МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ «МИСиС»

Институт компьютерных наук НИТУ МИСИС

Кафедра инженерной кибернетики

СБОРНИК СТАТЕЙ НАУЧНО-ТЕХНИЧЕСКОГО СЕМИНАРА СТУДЕНТОВ КАФЕДРЫ «ИНЖЕНЕРНОЙ КИБЕРНЕТИКИ» НА ТЕМУ «ИСКУССТВЕННЫЙ ИНТЕЛЛЕКТ В ПРОМЫШЛЕННЫХ, КОММЕРЧЕСКИХ, МЕДИЦИНСКИХ И ФИНАНСОВЫХ ПРИЛОЖЕНИЯХ»

Москва 2023

УДК 004.8
ББК 32.813.5

Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях, 2023: Сборник статей научно-технического семинара студентов. Вып. 1 / Под ред. А.Р. Ефимова— М.: НИТУ «МИСИС», 2023.— 168 с.: табл., ил., цв. ил.

Настоящий сборник содержит материалы научно-технического семинара «Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях», организатором которой является кафедра Инженерной кибернетики Института компьютерных наук НИТУ «МИСИС». На семинаре были представлены доклады по применению искусственного интеллекта в различных задачах народного хозяйства: промышленных, коммерческих, медицинских и финансовых приложениях.

Семинар проходил 30 декабря 2023 г. в режиме онлайн.

Содержание

Эмоциональный окрас комментариев на русском языке с помощью RuBERT моделей

Л. С. Измайлов
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1900850@edu.misis.ru

А. М. Устинов
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1902107@edu.misis.ru

***Аннотация* — в современном обществе активно обсуждается проблема эмоционального окраса комментариев на русскоязычных платформах, где пользователи выражают свои мнения и реагируют на различные события. Исследование фокусируется на применении RuBERT моделей для анализа эмоционального тона комментариев. Задачей является классификация комментариев на два основных типа: выражающие позитивное и негативное отношение. Модели глубокого обучения, основанные на RuBERT, предоставляют высокую точность в определении эмоционального содержания текста. В работе проводится сравнительный анализ двух RuBERT моделей, обученных на обширном корпусе комментариев, собранном с русскоязычных интернет-платформ. Полученные результаты могут быть полезны для создания инструментов управления контентом и обеспечения более позитивного взаимодействия пользователей в онлайн-среде.**

Ключевые слова — Обработка текста на естественном языке, Классификация комментариев, Анализ моделей, NLP, BERT, RuBERT, rubert-tiny2, rubert-base-cased

I. Введение

Обработка текста на естественном языке или же NLP (natural language processing) – это область, которая чаще всего лежит на стыке лингвистики, глубокого машинного обучения и искусственного интеллекта [1]. Обработка естественного языка сейчас не используется разве что в совсем устарелых отраслях NLP. Значительная часть данных в мире представлена в текстовом виде и данные могут иметь разнообразную структуру, информация может быть структурированной, не структурированной и частично структурированной. Методы обработки естественного языка успешно применяются во всех случаях. NLP включает в себя две важные области: NLU и NLG.

Для того чтобы работать с текстами на каком-либо языке, нужно знать о лингвистических феноменах, которые в этом языке есть, о лингвистических структурах и уметь переводить это в понятный язык для компьютеров. Основные направления, которые сейчас можно выделить в обработке и анализе естественных языков, это активное использование модели обучения без учителя и с учителем.

Современные технологии в области обработки естественного языка (NLP) включают в себя применение нейросетевых языковых моделей, таких как ELMo [2], BERT [3], и других, способных создавать контекстуальные векторные представления слов. Процесс обучения таких моделей проходит через два этапа: предобучение и дообучение. На этапе предобучения достигается изучение взаимосвязей между словами, в то время как этап дообучения обеспечивает эффективный перенос знаний для решения конкретной задачи.

Для продолжения развития в области моделей понимания языка требуются новые, более сложные задачи и соответствующие наборы данных. Среди таких задач можно выделить задачи классификации текста [4], вопросно-ответные системы [5], а также задачи выявления тональности текста [6] и другие подобные. В настоящее время доступны языковые модели, предназначенные для дообучения, специализирующиеся на анализе эмоционального окраса комментариев на русском языке. Необходимо, однако, отметить, что этап дообучения требует значительных вычислительных ресурсов.

II. Наборы данных

Для проведения дообучения и тестирования представленных в данном исследовании нейронных сетей были задействованы конкретные наборы данных. Рассмотрим более подробно открытые датасеты, которые были использованы в ходе исследования.

A. Kinopoisk-reviews

Датасет Кинопоиска (kinopoisk-reviews) [7] представляет собой крупный и широко используемый корпус данных в области обработки естественного языка (NLP) и машинного обучения. Он содержит более 80 тысяч отзывов о фильмах (рисунок 1), собранных с веб-сайта Кинопоиск, разбитых на два класса: положительные и отрицательные. На рисунке 2 представлены некоторые примеры комментариев, разбитые на эти классы.

Каждый отзыв представляет собой текстовое описание, включающее в себя мнение пользователя о фильме. Текст отзыва состоит в среднем из 2,5 тысяч символов или же 500 слов, если комментарий положительный и в около 2 тысяч символов или же 300 слов для негативных комментариев (рисунок 3). Эти тексты обычно предварительно обработаны, например, удалены стоп-слова, проведена лемматизация и токенизация.

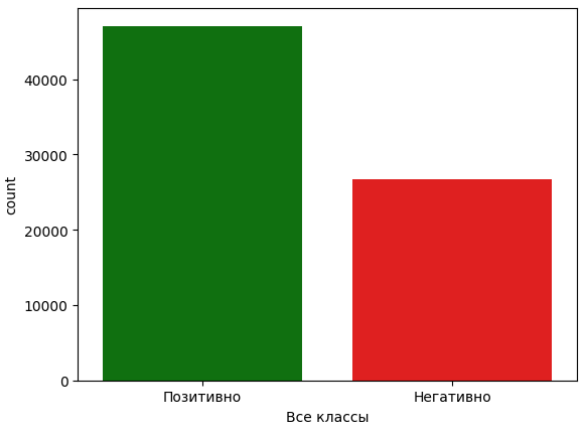


Рис. 1. Содержание датасета kinopoisk-reviews

text string	labels int64	label_name string
До просмотра этого фильма я считала К. Диаз неплохой комедийной актрисой, которая работает на приличном (во…	1	negative
Париж - город в котором сбываются мечты, в котором люди влюбляются, женятся и... ЛЮБЯТ!! Уезжая из этой страны…	1	negative
США времён Великой депрессии. Всё очень грустно, местами безжизненно, а главное бесчеловечно. В угоду…	0	positive
Я являюсь поклонником данного жанра и просмотрела довольно-таки много фильмов на данную тематику. Как бы…	1	negative
С начала я думал, что вот такое необычное вступление, а фильм оказался весь таким! Эта нудная вечеринка,…	1	negative
Прошло уже три года, с момента появления фильма 'сука-любовь'. На новую картину было выделено в десять раз…	0	positive

Рис. 2. Примеры отзывов, разбитых на классы

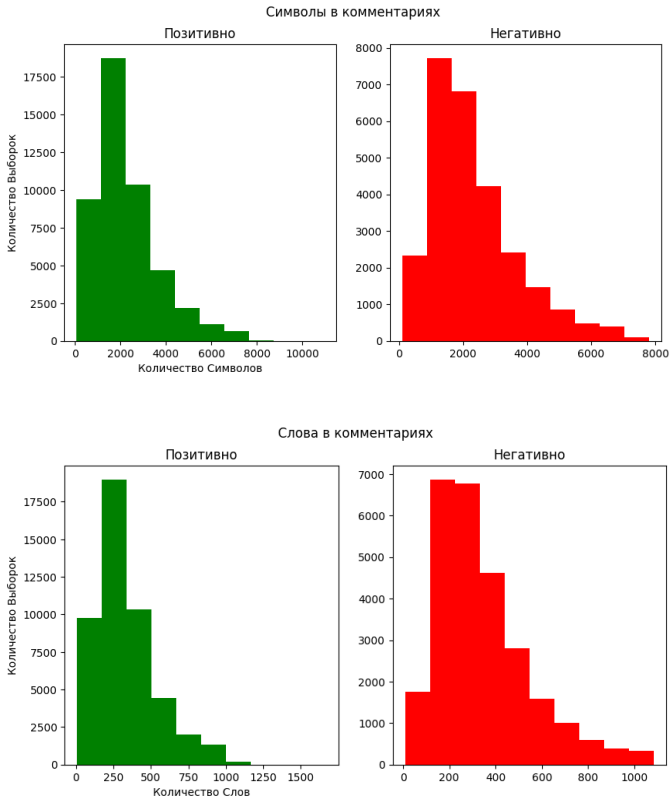


Рис. 3. Количество символов и слов в отзыве

B. Russian Language Toxic Comments

Набор данных Russian Language Toxic Comments [8] представлен в виде небольшой коллекции размером около 14 тысяч (рисунок 4) размеченных комментариев, собранных с известных русскоязычных онлайн-платформ 2ch.hk и pikabu.ru. Этот датасет создан с целью изучения токсичных языковых паттернов в русскоязычном онлайн-дискурсе и предоставляет ценный ресурс для исследования в области обработки естественного языка (Natural Language Processing, NLP) и анализа тональности текста в контексте онлайн-сообществ.

Данные представлены в виде комментариев с различных постов и блогов и также разделены на два класса. Текст комментария включает в себя около тысячи символов или же 100 слов для позитивных комментариев, а для негативных 500 символов или же 50 слов (рисунок 5). На рисунке 6 отображен пример разбиения этих комментариев.

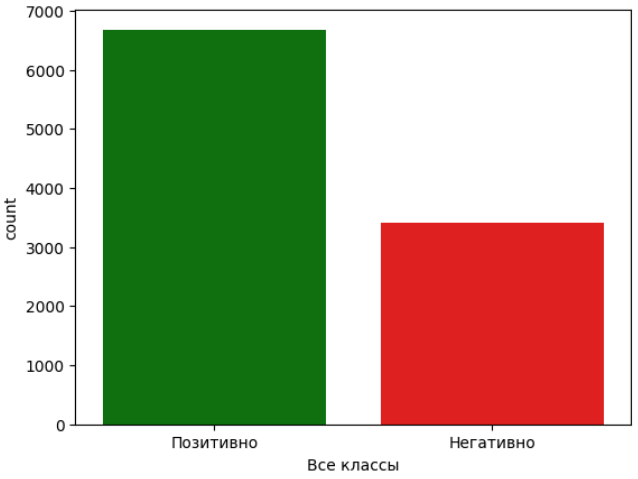


Рис. 4. Содержание датасета Russian Language Toxic Comments

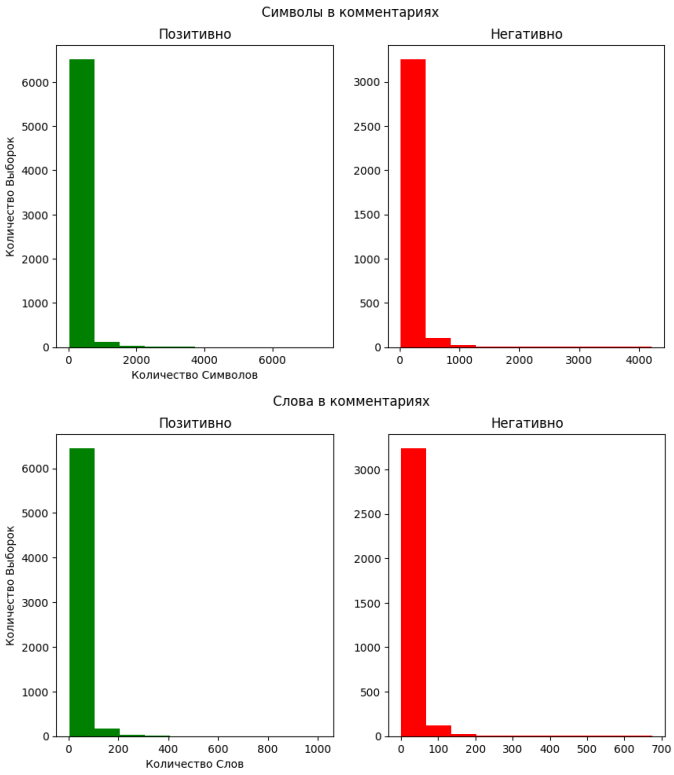


Рис. 5. Количество символов и слов в комментариях

comment	toxic
14412 unique values	0 1
С каких пор неадекват - это оскорбление? На Пикабу тысячи таких комментариев, и модераторы, даже буд...	1.0
50к в год это либо он тебе в уши ссыт чтобы ты не завидовал, либо я даже не знаю, в рабстве в каком-...	1.0
Возьмём как пример Россию, западноевропейские страны и США. Идёт метисация, сознательная политика за...	0.0
Может и старый, может и маразматик. Про то писать кириллицей или латинницей вам виднее, не спорю. Но...	0.0

Рис. 6. Примеры комментариев, разбитых на классы

C. *SearchTox*

На основе двух предыдущих датасетов был составлен собственный корпус, путем их объединения для более качественного исследования обработки естественного языка. На рисунках 7 и 8 представлена некоторая информация о датасете.

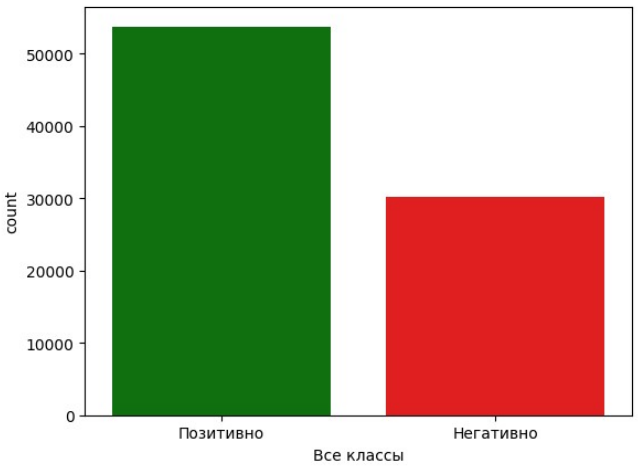


Рис. 7. Содержание датасета SearchTox

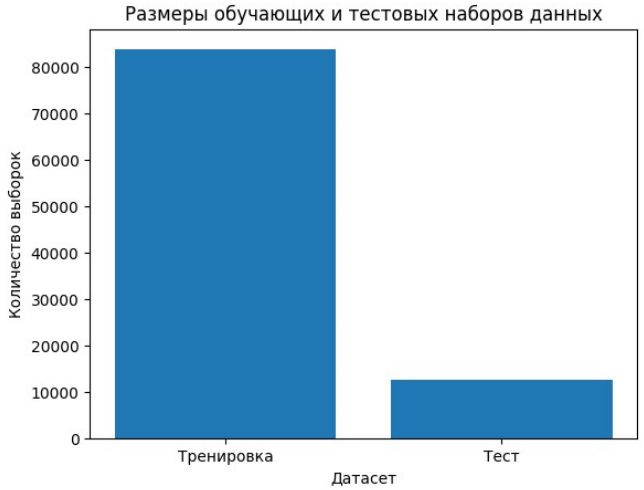


Рис. 8. Разбиение данных на тренировочную и тестовую выборку

III. Нейросетевые архитектуры

A. *BERT*

BERT – это метод, который использует предварительную подготовку языковых представлений [9]. Суть метода заключается в тренировке универсальной модели "понимания языка" на большом количестве текстов, а затем использовании этой модели для решения других задач в области обработки естественного языка, например, для поиска ответов на вопросы. По сравнению с предыдущими методами, BERT выделяется тем, что является первой неконтролируемой, глубоко двунаправленной моделью для предварительной подготовки языковых представлений.

Кроме того, BERT – это контекстуальная модель языка. Это означает, что при обработке текста BERT учитывает контекст, в котором находится каждое слово. Например, если слово "банк" используется в контексте "я пошел в банк", то оно будет иметь одно значение, а если в контексте "банк данных", то другое значение. Это помогает улучшить качество обработки текста и сделать его более точным.

BERT также обучается на больших объемах текстовых данных, что позволяет ему достигать высокого уровня точности в задачах обработки естественного языка. Например, с помощью BERT можно решать такие задачи, как классификация текстов, поиск ответов на вопросы, заполнение пропущенных слов и многое другое.

Концепция маскирования играет важную роль в архитектуре BERT и помогает модели понимать контекст, в котором находится каждое слово в предложении. В BERT используется два типа масок — маскирование слов и маскирование предложений.

Маскирование слов — это процесс, в котором модель скрывает некоторые слова в предложении и пытается предсказать их исходный токен. Для маскирования слов модель случайным образом заменяет часть слов в предложении на маскировочный токен [MASK], который является специальным символом в словаре модели BERT.

Например, рассмотрим предложение: "Я пошел в [MASK], чтобы купить молоко". Модель случайным образом выбирает одно из слов "магазин" или "парк" для

маскирования, и предложение становится: "Я пошел в [MASK], чтобы купить молоко". Модель теперь должна предсказать, что слово "магазин" должно быть заменено на [MASK].

Маскирование предложений — это процесс, в котором модель случайным образом скрывает одно из двух предложений, которые следуют друг за другом, в паре предложений. Затем модель должна предсказать, какое предложение было скрыто.

Например, рассмотрим пару предложений: "Кошки — это замечательные животные. Они могут быть очень ласковыми". Модель случайным образом выбирает одно из предложений для маскирования и предложение становится: "[MASK]. Они могут быть очень ласковыми". Модель теперь должна предсказать, что первое предложение было "Кошки — это замечательные животные".

Важно отметить, что модель BERT маскирует только около 15% слов в каждом предложении, чтобы предотвратить переобучение. Также, BERT использует механизмы самообучения и обучения с учителем, чтобы улучшить предсказание скрытых слов и предложений.

BERT использует специальный механизм взаимодействия между двумя предложениями, который называется механизмом NSP, чтобы понимать их взаимосвязь. Этот механизм помогает модели понимать, как два предложения связаны между собой, и учитывать эту связь при обработке естественного языка.

Механизм NSP работает следующим образом: перед тем как модель начинает обработку двух предложений, она добавляет специальный токен [CLS] в начало первого предложения и [SEP] в конец каждого предложения (рисунок 9). Затем модель применяет механизм маскирования для обоих предложений, а затем проходит через несколько слоев трансформации, чтобы получить представление о каждом предложении.

После этого модель использует выходные представления каждого предложения для прогнозирования, является ли второе предложение идущим за первым. То есть модель генерирует вероятность того, что второе предложение следует за первым (1) или не следует (0).

Использование BERT состоит из двух этапов: предварительного обучения и тонкой настройки.

Предварительное обучение BERT выполняется на огромном наборе данных, например, на 3,3 миллиарда слов из текстов Википедии и книг. Обучение выполняется на специализированном оборудовании, таком как GPU и TPU, чтобы ускорить процесс. Стоимость предварительного обучения BERT может достигать нескольких тысяч долларов, в зависимости от используемой аппаратуры и количества данных.

После предварительного обучения модель настраивается на конкретную задачу, используя процесс тонкой настройки. В этом процессе модель дополнительно обучается на небольшом наборе данных, связанных с конкретной задачей, такой как классификация текстов или поиск ответов на вопросы. Тонкая настройка обычно выполняется на более общедоступных серверах и может занять от нескольких часов до нескольких дней, в зависимости от объема обучающих данных и используемой аппаратуры.

Общие затраты на обучение модели BERT зависят от многих факторов, включая объем данных, используемую аппаратуру, продолжительность обучения и затраты на трудоемкий процесс подготовки данных. В целом, обучение модели BERT является трудоемким и затратным процессом, требующим большого количества вычислительных мощностей и времени. Однако, благодаря высокой точности и эффективности работы, BERT становится все более популярным инструментом в области обработки естественного языка.

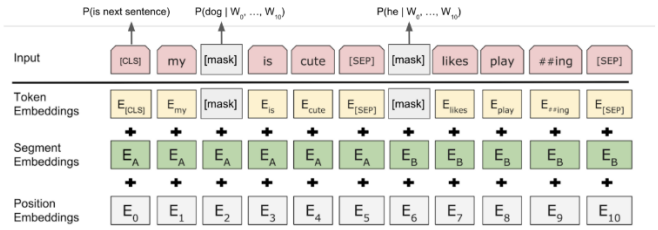


Рис. 9. Описание формата входных данных для модели BERT и процесс создания векторных представлений на основе этих данных

B. RuBERT

RuBERT [10] — это модель глубокого обучения для обработки естественного языка, предназначенная для работы с русским языком. Она является русской версией модели BERT.

RuBERT был обучен на большом наборе данных, состоящем из более чем 1,5 миллиарда слов, извлеченных из различных русскоязычных текстов, включая новостные статьи, литературные произведения, научные статьи и другие. Для обучения RuBERT использовались графические процессоры (GPU) и тензорные процессоры (TPU), чтобы ускорить процесс и снизить затраты на обучение.

Одна из особенностей RuBERT заключается в том, что она позволяет решать различные задачи, связанные с обработкой естественного языка, включая классификацию текстов, семантическую сегментацию, вопросно-ответную систему и другие. При этом модель предоставляет высокую точность и эффективность работы, что делает ее одним из наиболее востребованных инструментов для работы с русским языком.

IV. Дообучение

Процесс дообучения (fine tuning) в машинном обучении подразумевает использование заранее предобученной модели на новом наборе данных, специфичном для конкретной задачи.

Для процесса дообучения были выбраны две модели: rubert-tiny2 и rubert_base_cased_sentence. Для каждой из них были внесены дополнения в виде двух новых слоев.

Добавлен слой dropout с целью снижения вероятности переобучения модели.

Включен линейный слой с функцией активации гиперболического тангенса, предоставляющий два класса на выходе – положительный и негативный.

На рисунке 10 представлена схема слоев модели для дообучения.

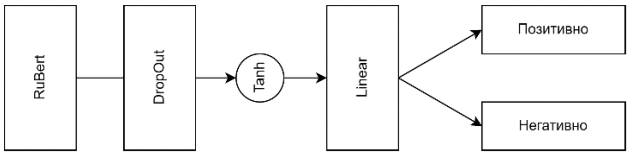


Рис. 10. Схема слоев модели для дообучения

V. Сравнение

Сравним две дообученные модели RuBERT, такие как rubert-tiny2 и rubert_base_cased_sentence. Исходя из ROC кривых, представленных на рисунках 11 и 12, можно сделать вывод, что обе модели были дообучены корректно.

Для оценки эффективности моделей мы использовали несколько метрик. Одной из наиболее распространенных является F1-мера (F1-score), которая является гармоническим средним между точностью (precision) и полнотой (recall).

TP (True Positive): количество комментариев, которые были правильно классифицированы как положительные.

FP (False Positive): количество комментариев, которые были ошибочно классифицированы как положительные (то есть модель сказала, что они положительные, но на самом деле они относятся к отрицательному классу).

FN (False Negative): количество комментариев, которые были ошибочно классифицированы как отрицательные (то есть модель сказала, что они отрицательные, но на самом деле они относятся к положительному классу).

Recall (Полнота): отношение TP к общему числу комментариев, которые действительно принадлежат к положительному классу. Полнота измеряет, насколько хорошо модель обнаруживает все положительные комментарии.

$$Recall = \frac{TP}{TP + FN} \quad (1)$$

Precision (Точность): отношение TP к общему числу комментарии, которые модель предсказала как положительные. Точность измеряет, насколько точными являются положительные предсказания модели.

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

F1 Score: это гармоническое среднее между точностью и полнотой. F1 Score высок, если и точность, и полнота высоки. F1 Score учитывает и точность, и полноту, предостерегая от переоценки модели, которая может быть высокой по одному из этих показателей, но низкой по-другому.

$$F1 = 2 * Precision * \frac{Recall}{Precision + Recall} \quad (3)$$

Таблица 1 отображает количественные оценки для двух моделей.

ТАБЛИЦА I. Оценка классификации двух моделей

	rubert-tiny2	**rubert_base_cased_sentence**
TP	7760	7854
FP	577	303
FN	358	264
Precision	0.93	0.96
Recall	0.96	0.97
F1	0.94	0.97
Время работы (секунд)	0.0064	0.01500

Исходя из таблицы 1 модель rubert_base_cased_sentence от разработчиков DeepPavlov показала себя лучше по сравнению с моделью rubert-tiny2, но стоит отметить, что дообучение модели rubert-tiny2 оказалось быстрее и ее размеры гораздо меньше, чем модель rubert_base_cased_sentence. На рисунках 13 и 14 отражены матрицы ошибок двух моделей.

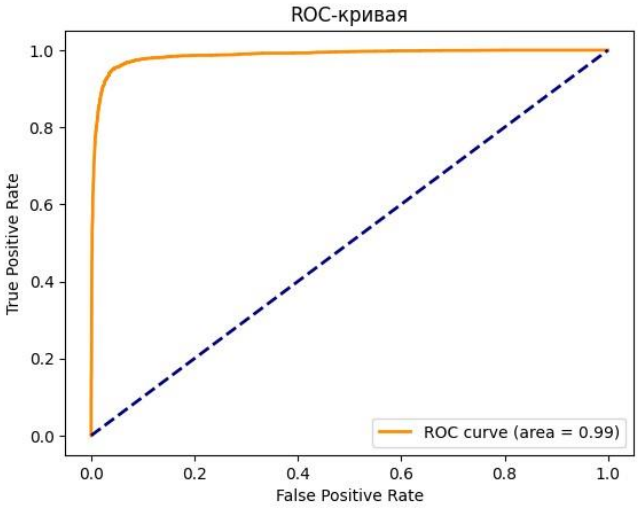


Рис. 11. ROC-кривая для модели rubert_base_cased_sentence

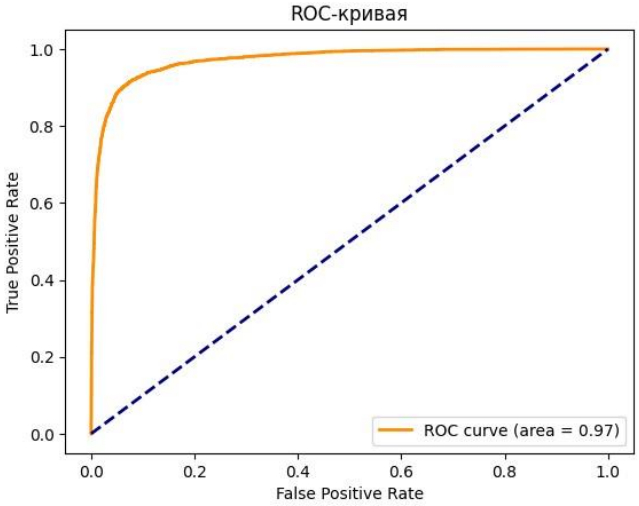


Рис. 12. ROC-кривая для модели rubert-tiny2

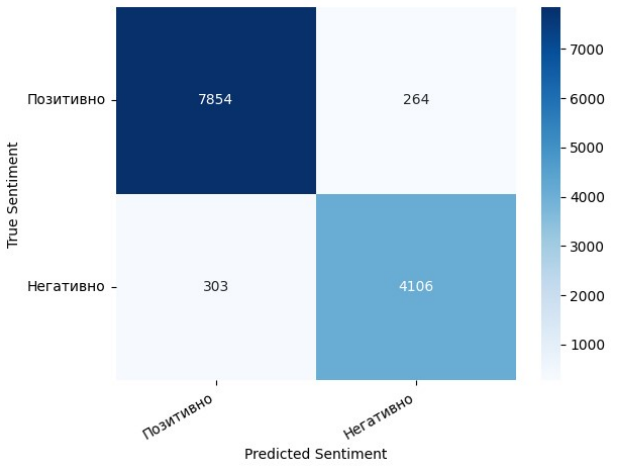


Рис. 13. Матрица ошибок для модели rubert_base_cased_sentence

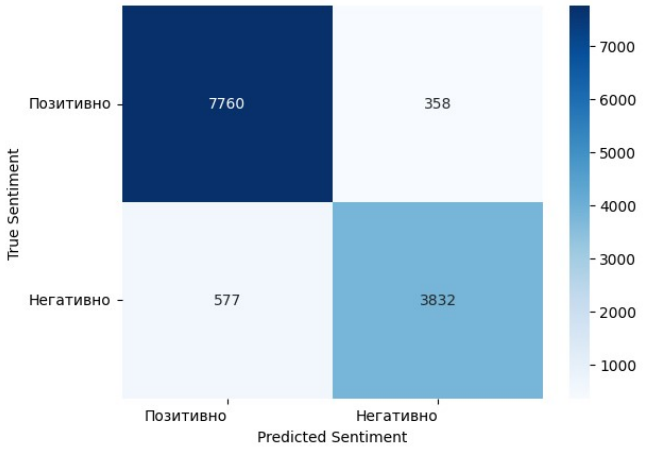


Рис. 14. Матрица ошибок для модели rubert-tiny2

VI. Заключение

Были рассмотрены и дообучены на собственном датасете две RuBERT модели: rubert_base_cased_sentence и rubert-tiny2. Исходя из полученных результатов при снятии метрик, стоит отметить, что выбор между моделями зависит от конкретной задачи. Если есть небольшой объем данных или ограничения по вычислительным ресурсам, rubert-tiny2 может быть более привлекательным вариантом из-за своей компактности и более быстрой инференсной скорости. Однако, если задача требует высокой точности и улучшенной производительности, rubert_base_cased_sentence может быть более предпочтительным выбором.

Литература

[1] Shervin Minaee, Nal Kalchbrenner, Erik Cambria, Narjes Nikzad, Meysam Chenaghlu, and Jianfeng Gao. 2021. Deep Learning--based Text Classification: A Comprehensive Review. ACM Comput. Surv. 54, 3, Article 62 (April 2022), 40 pages.

[2] Deep Contextualized Word Representations / M. Peters [и др.] // Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). — New Orleans, Louisiana : Association for Computational Linguistics, 06.2018. — С. 2227—2237.

[3] Bert: Pre-training of deep bidirectional transformers for language understanding / J. Devlin [и др.] // arXiv preprint arXiv:1810.04805. — 2018.

[4] Prabhu, S., Mohamed, M., & Misra, H. (2021). Multi-class text classification using BERT-based active learning. arXiv preprint arXiv:2104.14289.

[5] McCarley, J. S., Chakravarti, R., & Sil, A. (2019). Structured pruning of a bert-based question answering model. arXiv preprint arXiv:1910.06360.

[6] Лукашевич, Н. В. (2022). Автоматический анализ тональности текстов: проблемы и методы. *Интеллектуальные системы. Теория и приложения*, *26*(1), 50-61.

[7] kinopoisk-reviews dataset, available at: https://huggingface.co/datasets/zloelias/kinopoisk-reviews (Accessed: November 28, 2023).

[8] Russian Language Toxic Comments dataset, available at: https://www.kaggle.com/datasets/blackmoon/russian-language-toxic-comments/ (Accessed: November 28, 2023).

[9] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understandinga // Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. – 2019. – V.1 – P.4171–4186.

[10] Куратов Юрий Михайлович. Специализация языковых моделей для применения к задачам обработки естественного языка: автореферат дис. кандидата физико-математических наук: 05.13.17. // МФТИ — Московский физико-технический институт. 2020. URL: https://mipt.ru/upload/medialibrary/02f/avtoreferat-kuratov.pdf (дата обращения: 28.11.2023).

Методы машинного обучения для обнаружения вторжений в сетях интернета вещей

И. А. Антонов
кафедра инженерной кибернетики НИТУ МИСИС
Москва, Россия
m1908142@edu.misis.ru

М. К. Исаченко
кафедра инженерной кибернетики НИТУ МИСИС
Москва, Россия
m1904802@edu.misis.ru

***Аннотация*— интернет вещей (IoT) — это новая парадигма нашего времени, в которой интеллектуальные устройства и датчики со всего мира соединены в глобальную сеть, а распределенные приложения и услуги влияют на все сферы человеческой деятельности. Благодаря огромному экономическому эффекту и всепроникающему влиянию на нашу жизнь интернет вещей является привлекательной мишенью для преступников, а кибербезопасность становится приоритетом для экосистемы интернета вещей. Для защиты сетей интернета вещей от атак существуют системы обнаружения вторжений (IDS). В данной работе будут рассматриваться IDS, ядром которых являются методы машинного обучения, поскольку такие IDS способны к самообучению и могут работать при относительно небольших мощностях с достаточной скоростью, в отличие от классических IDS. В работе приведена исчерпывающая классификация атак на сети интернета вещей, рассмотрены методы классического машинного обучения и современные архитектуры нейронных сетей, в том числе, трансформерные модели, а также проведен сравнительный анализ их результатов применительно к задаче обнаружения вторжений в сетях IoT.**

Ключевые слова — интернет вещей, IoT, информационная безопасность, машинное обучение, нейронные сети, системы обнаружения вторжений, IDS.

I. Введение

С развитием интернета вещей (IoT) растет количество и разнообразие угроз безопасности компьютерных сетей [1]. Таким образом, обнаружение атак и защита сети стали очень сложной задачей для механизмов безопасности, таких как системы обнаружения вторжений (IDS) и системы предотвращения вторжений (IPS) [2]. Основные проблемы, возникающие при мониторинге и предотвращении этих атак, связаны с большим объемом данных, генерируемых устройствами IoT, а также с затратами времени и средств на их анализ и обработку. При большом количестве IoT-устройств и неоднородности сети атаки могут исходить из множества источников, генерируя постоянный поток данных, подлежащих анализу [3]. Это приводит к большой задержке в обнаружении атак и к увеличению количества ложных срабатываний, генерируемых текущими системами мониторинга [1].

Для решения данной задачи можно применить методы машинного обучения, поскольку в последние годы они активно развиваются и используются в различных областях от распознавания еды [4], до обнаружения ошибок проверки на полиграфе [5], генерации карт навигации [6], оценки качества зерна [7] и других. Машинное обучение позволяет компьютерным системам обучаться и принимать решения на основе данных, выявлять закономерности и строить прогнозы, невозможные для человека. Среди методов машинного обучения можно выделить классические методы, такие как решающие деревья и градиентный бустинг, а также более новые, например, глубокое обучение, генеративно-состязательные сети и трансформерные модели. Также методы машинного обучения были успешно применены в задаче обнаружения вторжений в программно-конфигурируемые сети [8], что является задачей, смежной с той, которая рассматривается в данной статье.

В связи с этим было предложено несколько методов машинного обучения (ML) для обнаружения присутствия вредоносных агентов в сети, что может означать возникновение атаки. В частности, методы глубокого обучения и методы, основанные на ансамблевом обучении, позволили добиться значительных результатов в плане точности и достоверности [9].

В данной работе мы рассматриваем 5 методов классического машинного обучения и 5 архитектур нейронных сетей и проводим сравнительный анализ их производительности на части современного датасета, собранного в 2023 году.

II. Сети интернета вещей

A. *Архитектура интернета вещей*

В настоящее время во всем мире наблюдается недостаток согласованности и стандартизации в решениях интернета вещей, из-за чего возникают проблемы, связанные с совместимостью и управляемостью такими сетями [10]. Считается, что из-за этого отсутствия стандартизации мир до сих пор не смог договориться о единой эталонной модели IoT [11]. Для унификации в статье [12] предлагаются обобщенная архитектура интернета вещей и многоуровневый стек протоколов интернета вещей.

Экосистема интернета вещей может включать в себя различные типы устройств, которые могут быть развернуты в любой из следующих топологий: звезда, кластерное дерево и ячеистая сеть. «Вещи» обычно подключаются к шлюзовому устройству с использованием различных протоколов связи IoT, таких как 802.15.4, LoRaWAN, SigFox, ZigBee, WiFi, Bluetooth Low Energy (BLE), Near Field Communication и радиочастотная идентификация (RFID). Устройство-шлюз подключается к приложению или сетевому серверу через 3G/4G, LTE (Long-Term Evolution), оптоволоконный кабель (OFC), спутниковую связь и т. д. Серверы сети/приложений (могут быть расположены в облаке) предоставляют различные услуги по анализу данных своим пользователям и третьим лицам, включая государственные и частные организации. Обработанные данные превращаются в полезную информацию в виде статистики здравоохранения, автономных

служб умного дома, бизнес-аналитики, промышленной автоматизации, экологического мониторинга, пригодных для жизни городских сообществ и услуг совместного использования умных городов.

Что касается стека протоколов интернета вещей, первый уровень – это физический уровень, который состоит из датчиков, исполнительных механизмов, вычислительного оборудования, идентификации и адресации вещей. Его цель — воспринимать данные из окружающей среды. Весь сбор и обработка данных осуществляется на этом уровне. Некоторые другие функции физического уровня включают выбор частоты, модуляцию-демодуляцию, шифрование-дешифрование, передачу и прием данных. Проблемы, с которыми сталкивается этот уровень, — энергопотребление, безопасность и совместимость. Второй уровень — это сетевой уровень, который отвечает за получение данных от сенсорных устройств и последующую отправку их на уровень приложений для обработки, аналитики и интеллектуальных услуг. Сетевой уровень также сталкивается с проблемами, касающимися масштабируемости, доступности сети, энергопотребления и безопасности. Третий уровень — это уровень приложений, который предоставляет клиентам интеллектуальные услуги, а также передает обработанные/агрегированные данные на семантический уровень. Проблемы, с которыми сталкиваются на этом уровне, связаны с хранением и обработкой данных, полученных от датчиков, безопасностью/конфиденциальностью пользовательской информации и соответствием промышленным/правительственным нормам. Четвертый и последний уровень — семантический, который также можно назвать уровнем управления бизнесом, поскольку он управляет всеми действиями системы интернета вещей. Это подразумевает использование когнитивных технологий для предоставления определенных высококачественных услуг, таких как анализ данных, бизнес-аналитика, принятие стратегических решений и бизнес-моделирование.

B. Отличия IoT и традиционных сетей

Рассмотрим теперь отличия интернета вещей от традиционных сетей [12].

Интернет вещей обычно включает в себя встроенные устройства с ограничениями ресурсов, такие как RFID и сенсорные узлы. Эти устройства имеют небольшой объем памяти, низкую вычислительную мощность, небольшой объем дискового пространства и требуют низкого энергопотребления. В это же время, что традиционные сети состоят из компьютеров, серверов и смартфонов, обладающих большим количеством ресурсов. Следовательно, традиционные сети могут поддерживаться сложными и многофакторными протоколами безопасности без учета ресурсов. В отличие от этого, системы интернета вещей требуют облегченных алгоритмов безопасности, которые поддерживают баланс между безопасностью и потреблением ресурсов, таких как время автономной работы.

Устройства интернета вещей в основном подключаются к Интернету или шлюзовым устройствам через более медленные и менее безопасные беспроводные среды связи, такие как 802.15.4, 802.11a/b/g/n/p, LoRa, ZigBee, NB-IoT и SigFox. В результате системы интернета вещей склонны к утечке данных и другим проблемам конфиденциальности. В традиционном же Интернете конечные устройства обмениваются данными через более безопасные и быстрые проводные/беспроводные среды, такие как оптоволокно, DSL/ADSL, Wi-Fi, 4G и LTE. Другое отличие состоит в том, что традиционные сетевые устройства имеют почти одни и те же ОС и формат данных, но в случае интернета вещей из-за специфичных для приложений функций и отсутствия ОС содержимое и форматы данных различаются. Следовательно, из-за такого разнообразия сложно разработать стандартный протокол безопасности, подходящий для всех типов устройств и систем интернета вещей. В результате широкий спектр угроз интернета вещей по-прежнему существует и угрожает безопасности и конфиденциальности пользователей.

Если рассмотреть структуру безопасности, традиционные сети защищены сочетанием статической защиты периметра сети на основе межсетевых экранов, IDS/IPS, а конечные устройства защищены хост-подходами такими как антивирус и исправления безопасности/программного обеспечения. Хост-решения не могут быть применены к устройствам IoT из-за ограниченности ресурсов. Аналогичным образом, традиционный механизм защиты периметра не может защитить устройства IoT, поскольку эти устройства развернуты глубоко в сети. Следовательно, устройства IoT не могут быть защищены только с помощью хост-решений.

III. Атаки на сети IoT

Авторы статьи [12] предлагают следующую классификацию атак на сети IoT.

A. Общие атаки

Маскарадинг и несанкционированное раскрытие личной информации. Почти 90% устройств в той или иной форме собирают личную информацию о пользователях. Такое несанкционированное хранение информации уязвимо для атак на безопасность данных, конфиденциальность и целостность.

Атаки на целостность устройств. Конечные устройства IoT в основном работают в ненадежной среде без какой-либо физической защиты. Следовательно, эти устройства подвергаются физическим атакам, включая инвазивные аппаратные атаки, атаки по побочным каналам и атаки обратного проектирования. Подобные взломанные физическим путем IoT-устройства могут использоваться для осуществления других более сложных атак таких как Mirai, DDoS, за счет контроля злоумышленником отправляемых с устройства сетевых запросов.

Удаленное выполнение кода. Ошибки при выделении памяти могут быть использованы для выведения системы из строя и удаленного выполнения вредоносного кода в уязвимых системах IoT.

Атака на целостность ПО/кода. Целостность программного обеспечения, включая целостность операционной системы, приложений и конфигураций устройств интернета вещей, является ключевым элементом, гарантирующим безопасность и конфиденциальность «вещей». Поэтому устройства IoT необходимо защищать от атак вредоносного ПО, таких как трояны, вирусы и другие атаки во время выполнения.

Атаки на протоколы коммуникации. Большинство современных беспроводных коммуникаций

придерживаются многоуровневой архитектуры протоколов OSI, а шифрование физического уровня не снабжается дополнительными механизмами безопасности на верхних уровнях связи.

DoS и DDoS. Из-за ограничений ресурсов, таких как нехватка памяти, низкая вычислительная мощность и низкое потребление батареи, устройства IoT уязвимы для атак на истощение ресурсов. Эти атаки включают в себя глушение каналов связи, масштабные несанкционированные или злонамеренное использование критически важных ресурсов интернета вещей, таких как пропускная способность, память, время процессора, дисковое пространство и изменение конфигурации узла. Все эти атаки, могут повлиять на работоспособность IoT-устройств и недоступность их сервисов для соответствующих пользователей.

B. Атаки на физический уровень

Подслушивание. Злоумышленники могут устанавливать устройства, аналогичные конечным узлам интернета вещей для прослушивания беспроводного трафика и извлечения ценной информации о пользователях.

Потеря питания / разряд аккумулятора. Атака разряда батареи подвергает узел большому количеству легальных запросов, что не позволяет ему перейти в спящий режим или режим энергосбережения.

Внедрение вредоносных данных. Злоумышленники могут ввести ложные данные, что приведет к неадекватной или опасной реакции системы [13].

Атака Сивиллы. Узел злоумышленника может создать несколько поддельных идентификаторов, что может повлиять на результаты системы отказоустойчивости на основе голосования или протокола маршрутизации.

Атаки по побочным каналам. Эти атаки основаны на информации побочного канала об устройстве шифрования.

Взлом устройства. Получение доступа к устройству и полного контроля над ним.

Атаки по времени. Порты отладки (UART (универсальный асинхронный приемник-передатчик), JTAG и т. д.), оставленные производителями открытыми, делают систему уязвимой для таймингових атак и перепрошивки внешней памяти.

Полуинвазивные и инвазивные вторжения. Инвазивные атаки требуют физического доступа к системе и предполагают нарушение ее целостности для получения информации (например, вскрытие корпуса). Полуинвазивные атаки также требуют доступа к устройству, но при этом не нарушают целостности, а используют другие методы воздействия (например, с помощью лазерного луча).

Изменение конфигурации/версии прошивки. Неправильная реализация функций шифрования и хеширования угрожает безопасности базовой системы.

Несанкционированный доступ к устройствам. Использование пользователями паролей по умолчанию и жестко запрограммированные производителями имена пользователей и пароли являются серьезной уязвимостью безопасности.

C. Атаки на сетевой уровень

DoS-атаки. Коллизионные атаки, атаки перегрузки канала, разрядка батареи; отправка поддельных/ложных сообщений на узел, сервер или шлюзовое устройство.

Атака фрагментации. Фрагментация IP-адресов для нарушения работы сервисов или отключения устройств.

Атака посредника (MITM). Когда часть информации запрашивается из Интернета, она проходит через несколько маршрутизаторов, прежде чем достигнет пункта назначения. Злоумышленник тайно встает между пользователем и Интернетом и может прочитать всю информацию, которая проходит через маршрутизаторы, если данные не зашифрованы [14].

Спуфинг, hello flood и homing атаки. Во время атаки IP Spoofing злоумышленник пытается имитировать поведение другого устройства, подделывая IP-адрес в заголовке пакета при подключении к Интернету.

Атаки фабрикации / модификации / воспроизведения сообщений. Атаки, основанные на манипуляциях с управляющими сообщениями.

Вторжение в сеть и компрометация устройства. Происходит удаленно, с использованием вредоносного ПО.

Атака репликации узла и внедрение мошеннического устройства. Злоумышленник может внедрить поддельное устройство в систему интернета вещей, чтобы подслушивать радиотрафик, внедрять сфабрикованные сообщения или наводнять радиоканалы фальшивыми сообщениями, чтобы сделать систему недоступной для законных пользователей.

Атаки на хранилища. Большие объемы данных, содержащие важную информацию о пользователе, необходимо хранить на устройствах хранения данных или в облаке. И то, и то можно атаковать, и данные могут быть скомпрометированы или изменены в деталях [13].

D. Атаки на уровень приложений

Вредоносные коды. Внедрение вредоносного кода предполагает изменение строк исходного кода и замену его кодом, который может повредить систему пользователя. Вредоносный код внедряется в приложение носимого устройства, что позволяет злоумышленнику воспользоваться уязвимостью системы [14].

Модификация программного обеспечения. Злоумышленник может скомпрометировать модифицировать программное обеспечение или встроенное ПО для выполнения несанкционированных действий.

Брутфорс и атаки по словарю, повышение привилегий и подделка данных. Злоумышленник угадывает учетные данные по умолчанию для некоторых приложений или учетные данные (лицензии) интеллектуальных устройств [15].

SQL-инъекции. Использование вредоносного кода на языке SQL для манипулирования базой данных и получения доступа к потенциально ценной информации.

Кража личных данных и компрометация пароля/ключа/токена сеанса. Злоумышленник может

извлечь список авторизованных пользователей и позже замаскироваться под законного пользователя.

Раскрытие конфиденциальных/частных данных. Злоумышленник может перехватить связь между пользователем и умным устройством и выяснить личные привычки пользователя.

XSS (межсайтовый скриптинг). Такая уязвимость позволяет злоумышленнику запустить произвольный код JavaScript в браузер жертвы. В дальнейшем это может привести к взлому телефона и краже личных данных.

E. Атаки на семантический уровень

На этом уровне чаще всего происходят атаки кражи личных данных и нарушения конфиденциальности пользователей.

IV. Набор данных

Для обучения и тестирования рассматриваемых в данной работе алгоритмов машинного обучения использовался набор данных *CIC IoT Dataset 2023* [16], который был собран путем совершения 33-х атак в IoT топологии. Эти атаки подразделяются на семь категорий, а именно: *DDoS, DoS, Recon, Web-based, brute force, spoofing и Mirai*. При этом все атаки выполняются вредоносными IoT-устройствами, нацеленными на другие IoT-устройства.

IoT-топология, развернутая для создания *CIC IoT 2023*, показана на рисунке 1 и включает 105 IoT-устройств. Всего в атаках было непосредственно задействовано 67 IoT-устройств, а еще 38 устройств Zigbee и Z-Wave были подключены к пяти хабам.

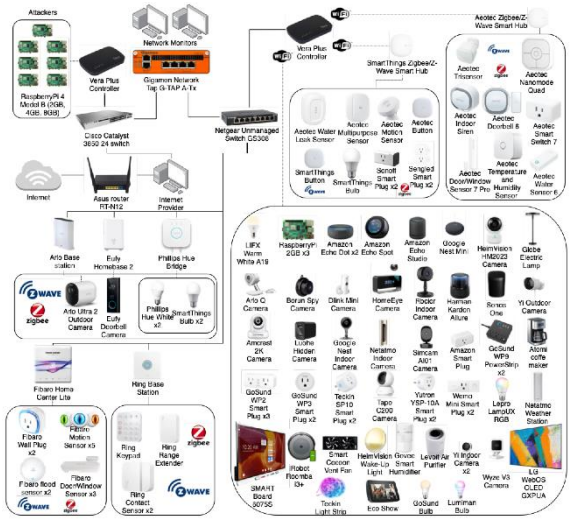


Рис. 1. Топология IoT-сети, использованная в экспериментах

Для каждой атаки проводился отдельный эксперимент, нацеленный на все применимые устройства. Во всех сценариях атаки осуществлялись вредоносными устройствами интернета вещей, нацеленными на уязвимые устройства интернета вещей. Например, DDoS-атаки выполнялись против всех устройств, тогда как веб-атаки были нацелены на устройства, поддерживающие веб-приложения. На рисунках 2 и 3 показано количество случаев для каждой атаки и категории.

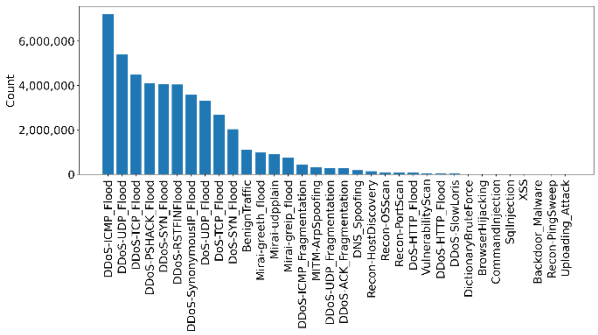


Рис. 2. Количество записей для каждого сценария

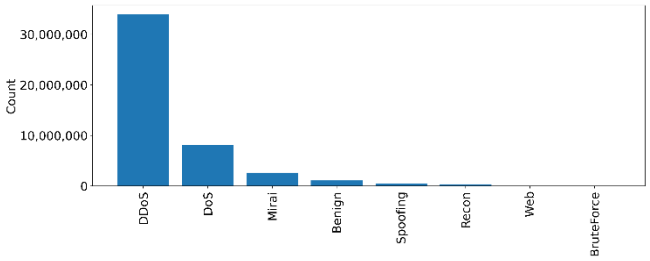


Рис. 3. Количество записей для каждой категории

В таблице 1 представлены классы атак, использованные в работе, и количество объектов каждого класса.

ТАБЛИЦА I. Классы атак датасета

Класс	**Количество объектов в датасете**
Нормальный	1 098 195
DDoS	33 984 560
DoS	8 090 738
Mirai	2 634 124
Spoofing	486 504
Recon	354 565
Web	24 829
BruteForce	13 064

Ввиду очень большого количества записей в датасете, его несбалансированности и ограниченности доступных вычислительных ресурсов, было принято решение использовать в данной работе не весь набор данных, а только его часть, содержащую по 13 064 объекта каждого класса.

В результате, датасет, использованный в работе, представляет из себя таблицу, в которой каждая строка описывает один пакет трафика сети IoT, а каждый столбец – определенный признак этого трафика.

V. Проведенные исследования и результаты

A. Предобработка датасета и выбор функций оценки

Первым этапом работы стала предобработка датасета для лучшей работы алгоритмов машинного обучения. Для этого были произведены следующие действия:

1. Из 46 признаков в датасете оставлено только 43, не содержащие NaN-значений.
2. Была выделена целевая переменная (создан отдельный массив, содержащий только метки классов трафика).

3. Датасет был нормализован для поддержания баланса между влиянием входных переменных, представленных в разных масштабах, на выходную переменную и, как следствие, улучшения работы алгоритмов машинного обучения.
4. Датасет был разделен на тренировочную, тестовую и валидационную выборки в соотношении 80%-10%-10%.

После этого модели машинного обучения были обучены на тренировочной выборке и проверены на тестовой.

Для оценки качества моделей были использованы меры *accuracy* и F_1. Значение *accuracy* — это отношение количества правильно классифицированных объектов к общему количеству элементов выборки. Значение F_1-меры есть среднее гармоническое двух других функций оценки качества: *precision* (доля объектов, действительно принадлежащих данному классу относительно всех объектов, которые модель отнесла к этому классу) и *recall* (доля найденных классификатором объектов, принадлежащих классу, относительно всех объектов этого класса). F_1-мера объединяет в себе две данных оценки, облегчая тем самым понимание, насколько качественно работает модель.

B. Использованные модели классического ML

В работе были использованы следующие модели классического машинного обучения.

Decision Tree. Модель решающего дерева была взята из библиотеки SciKit Learn [17] с параметрами: {criterion: enthropy; max_depth: 20; min_samples_split: 38}.

Random Forest. Модель случайного леса была взята из библиотеки SciKit Learn [17] с параметрами: {criterion: gini; max_depth: 40; max_features: sqrt; n_estimators: 300}.

k-Nearest Neighbors. Модель *k* ближайших соседей была взята из библиотеки SciKit Learn [17] с параметрами: {leaf_size: 40; metric: euclidean; n_neighbors: 4}.

Logistic Regression. Модель логистической регрессии была взята из библиотеки SciKit Learn [17] с параметрами: {C: 10000; max_iter: 300; multi_class: ovr}.

XGBoost. Модель градиентного бустинга была взята из библиотеки XGBoost [18] с параметрами: {max_depth: 20; eta: 0,3; silent: 1; objective: multi:softprob; num_class: 8}.

Для всех перечисленных методов кроме XGBoost гиперпараметры были подобраны при помощи метода поиска по сетке, реализация которого также есть в SciKit Learn [17].

C. Использованные нейронные сети

В работе были использованы следующие нейронные сети.

DatRet. Новая архитектура DatRet, разработанная в 2023 году для работы с табличными данными [19], представляет из себя архитектуру MLP, но в ней количество скрытых слоев и нейронов на каждом из них генерируется автоматически, в зависимости от указанного количества нейронов на первом скрытом слое. В данной работе была использована с параметрами по умолчанию из соответствующей библиотеки языка Python.

Сверточная нейронная сеть (CNN). Одномерная сверточная нейронная сеть была построена с помощью библиотеки TensorFlow [20, 21] и имеет следующую архитектуру:

1. Одномерный сверточный слой: 128 фильтров, размер ядра 3, функция активации ReLU.
2. Одномерный сверточный слой: 64 фильтра, размер ядра 3, функция активации ReLU.
3. Слой Dropout с параметром 0,5.
4. Слой Max Pooling с размером пула 2.
5. Слой Flatten.
6. Полносвязный слой, 128 нейронов, функция активации ReLU.
7. Полносвязный выходной слой, 8 нейронов, функция активации SoftMax.

Функция потерь – категориальная кросс-энтропия, метод оптимизации – ADAM, количество эпох – 50.

LSTM. Сеть долгой краткосрочной памяти была построена с помощью библиотеки TensorFlow [20, 21] и имеет следующую архитектуру:

1. Входной слой формы (None, 42).
2. Слой LSTM, количество ячеек 256, функция активации: ReLU.
3. Слой LSTM, количество ячеек 128, функция активации: ReLU.
4. Слой LSTM, количество ячеек 128, функция активации: ReLU.
5. Слой LSTM, количество ячеек 128, функция активации: ReLU.
6. Полносвязный слой, 100 нейронов, функция активации: ReLU.
7. Полносвязный слой, 80 нейронов, функция активации: ReLU.
8. Полносвязный слой, 8 нейронов, функция активации: Softmax.

Функция потерь – категориальная кросс-энтропия, метод оптимизации – ADAM, количество эпох – 30.

TabNet. TabNet [22] – новая архитектура полносвязной нейронной сети, разработанная специально для работы с табличными данными, имеющая следующие особенности:

- TabNet принимает на вход табличные данные без какой-либо предварительной обработки и обучается с использованием оптимизации на основе градиентного спуска;
- TabNet использует, наряду с полносвязными слоями для табличных данных, механизм внимания, который позволяет интерпретировать ее результат.

Таким образом, данная архитектура совмещает в себе производительность нейронных сетей и интерпретируемость методов, основанных на решающих деревьях.

TabPFN. TabPFN [23] – обученный трансформер, который может выполнять классификацию небольших наборов табличных данных менее чем за секунду, не требует настройки гиперпараметров и конкурирует с современными методами классификации. TabPFN выполняет контекстное обучение (ICL), он учится делать прогнозы, используя последовательности помеченных примеров (x, $f(x)$), заданных на входе, без необходимости дальнейшего обновления параметров.

Для данной модели была сформирована отдельная выборка из датасета, содержащая по 300 объектов каждого класса, поскольку для ее работы требуется очень большое количество вычислительных ресурсов.

D. Результаты

В таблице 2 представлены результаты работы всех методов машинного обучения, использованных в работе.

ТАБЛИЦА II. Полученные результаты

Модель	***Accuracy***	***F*$_1$-мера**
Decision Tree	0.893	0.892
Random Forest	0.906	0.906
k-Nearest Neighbors	0.711	0.710
Logistic Regression	0.644	0.644
XGBoost	0.912	0.911
DatRet	0.753	0.753
CNN	0.756	0.756
LSTM	0.732	0.734
TabNet	0.698	0.697
TabPFN	0.700	0.705

Мы видим, что лучше всех с задачей справилась модель градиентного бустинга. Среди нейронных сетей лучший результат показала сверточная нейронная сеть. Покажем для данных моделей матрицы ошибок (рис. 4 и 5 соответственно) и динамику обучения для CNN (рис. 6–7).

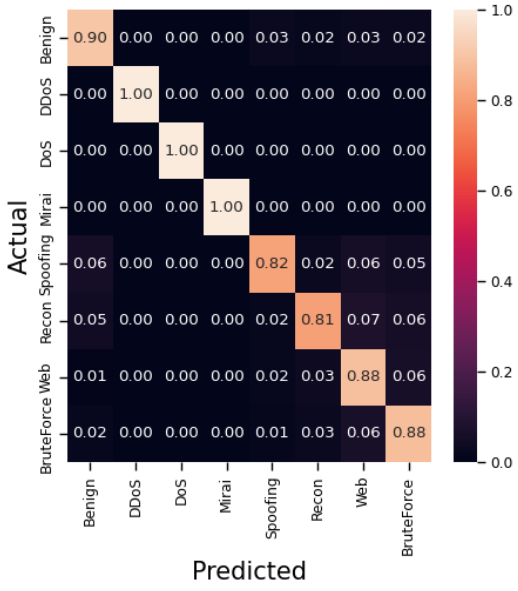


Рис. 4. Матрица ошибок для XGBoost

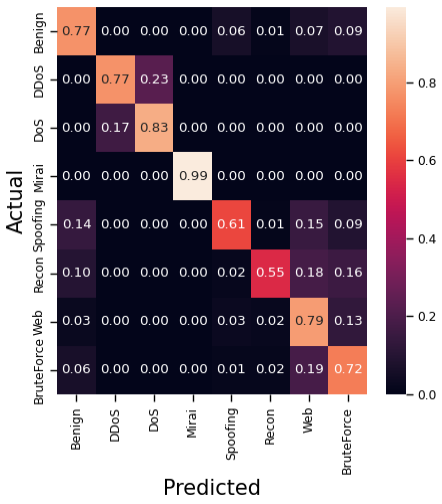


Рис. 5. Матрица ошибок для CNN

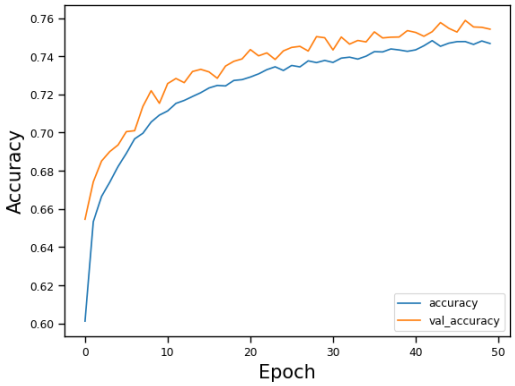


Рис. 6. Зависимость *accuracy* от количества эпох на обучающей (синий) и валидационной (оранжевый) выборках для CNN

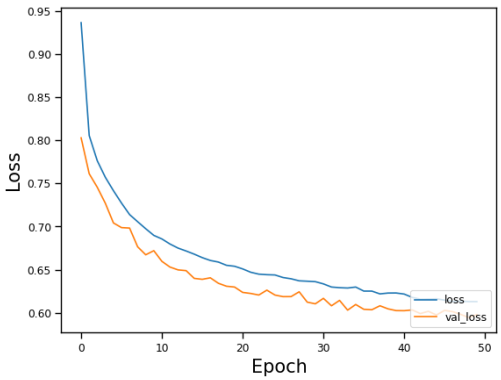


Рис. 7. Зависимость потерь от количества эпох на обучающей (синий) и валидационной (оранжевый) выборках для CNN

VI. Заключение

В процессе работы была составлена классификация атак на сети интернета вещей, после чего выбран актуальный (2023) набор данных для решения задачи обнаружения вторжений в сетях IoT.

Были реализованы различные модели машинного обучения для решения задачи классификации трафика в сетях IoT и проведен сравнительный анализ результатов их работы.

Для моделей классического машинного обучения (кроме XGBoost) был проведен подбор гиперпараметров, что позволило улучшить качество их работы.

В работе было использовано несколько новых архитектур нейронных сетей: архитектура DatRet, разработанная в 2023 году для работы с табличными данными; архитектура TabNet, представленная в 2021 году; архитектура TabPFN 2022 года.

Лучший результат решения задачи показала модель XGBoost. Лучший результат среди нейронных сетей показала модель CNN.

В дальнейшем планируется решать данную задачу, задействуя большее количество вычислительных мощностей. Это даст возможность проводить исследования на полном датасете, содержащем более 1 млрд записей, а также более гибко подбирать гиперпараметры для моделей.

Литература

[1] X. Liang and Y. Kim, “A survey on security attacks and solutions in the iot network,” in 2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC). IEEE, 2021, pp. 0853–0859.

[2] P. R. Maidamwar, M. M. Bartere, and P. P. Lokulwar, “Implementation of network intrusion detection system using artificial intelligence: Survey,” in Proceedings of the 2nd International Conference on Recent Trends in Machine Learning, IoT, Smart Cities and Applications. Springer, 2022, pp. 185–198.

[3] A. G. P. Lobato, M. Andreoni Lopez, and O. Duarte, “Um sistema acurado de detecc¸ao de ameac¸as em tempo real por processamento ˜ de fluxos,” XXXIV Simposio Brasileiro de Redes de Computadores e ´ Sistemas Distribu´ıdos-SBRC, 2016.

[4] Kudryashov A. A., Mishchanin M. A., Sadekov R. N. Food recognition using deep learning networks and order history for smart canteen checkout automation.

[5] Asonov D. et al. Building a second-opinion tool for classical polygraph //Scientific Reports. – 2023. – Т. 13. – №. 1. – С. 5522.

[6] Sadekov R. N. et al. Road sign detection and recognition in panoramic images to generate navigational maps //2017 24th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS). – IEEE, 2017. – С. 1-5.

[7] Minkin U. I. et al. Computer vision system: a tool for evaluating the quality of wheat in a grain tank //Tenth International Conference on Machine Vision (ICMV 2017). – SPIE, 2018. – Т. 10696. – С. 451-457.

[8] Kurochkin I. I., Volkov S. S. Using GRU based deep neural network for intrusion detection in software-defined networks //IOP Conference Series: Materials Science and Engineering. – IOP Publishing, 2020. – Т. 927. – №. 1. – С. 012035.

[9] A. Salih, S. T. Zeebaree, S. Ameen, A. Alkhyyat, and H. M. Shukur, “A survey on the role of artificial intelligence, machine learning and deep learning for cybersecurity attack detection,” in 2021 7th International Engineering Conference “Research & Innovation amid Global Pandemic”(IEC). IEEE, 2021, pp. 61–66.

[10] Banafa A. IoT standardization and implementation challenges //IEEE internet of things newsletter. – 2016. – Т. 2016. – С. 1-10.

[11] Al-Fuqaha A. et al. Internet of things: A survey on enabling technologies, protocols, and applications //IEEE communications surveys & tutorials. – 2015. – Т. 17. – №. 4. – С. 2347-2376.

[12] Makhdoom I. et al. Anatomy of threats to the internet of things //IEEE communications surveys & tutorials. – 2018. – Т. 21. – №. 2. – С. 1636-1675.

[13] Kumar S. A., Vealey T., Srivastava H. Security in internet of things: Challenges, solutions and future directions //2016 49th Hawaii International Conference on System Sciences (HICSS). – IEEE, 2016. – С. 5772-5781.

[14] Shah Y., Sengupta S. A survey on Classification of Cyber-attacks on IoT and IIoT devices //2020 11th IEEE Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON). – IEEE, 2020. – С. 0406-0413.

[15] Mann P. et al. Classification of various types of attacks in IoT environment //2020 12th International Conference on Computational Intelligence and Communication Networks (CICN). – IEEE, 2020. – С. 346-350.

[16] Neto E. C. P. et al. CICIoT2023: A real-time dataset and benchmark for large-scale attacks in IoT environment. – 2023.

[17] Pedregosa et al. Scikit-learn: Machine Learning in Python, JMLR 12, pp. 2825-2830, 2011.

[18] Tianqi Chen and Carlos Guestrin. XGBoost: A Scalable Tree Boosting System. In 22nd SIGKDD Conference on Knowledge Discovery and Data Mining, 2016

[19] DatRet: Tensorflow implementation for structured tabular data // Medium URL: https://medium.com/mlearning-ai/datret-tensorflow-implementation-for-structured-tabular-data-9dd7ff71bcd1 (доступ 20.11.2023).

[20] Abadi M. et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems //arXiv preprint arXiv:1603.04467. – 2016.

[21] Abadi M. et al. {TensorFlow}: a system for {Large-Scale} machine learning //12th USENIX symposium on operating systems design and implementation (OSDI 16). – 2016. – С. 265-283.

[22] Arik S. Ö., Pfister T. Tabnet: Attentive interpretable tabular learning //Proceedings of the AAAI conference on artificial intelligence. – 2021. – Т. 35. – №. 8. – С. 6679-6687.

[23] Hollmann N. et al. Tabpfn: A transformer that solves small tabular classification problems in a second //arXiv preprint arXiv:2207.01848. – 2022.

Исследование возможности классификации мусора при помощи компьютерного зрения

И. И. Антипов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2306246@edu.misis.ru

***Аннотация* — в настоящее время проблема утилизации мусора становится все более актуальной в современном обществе, а эффективные методы классификации отходов могут помочь в автоматизации процесса его переработки. Цель проведения данного исследования – ответить на вопрос, достаточно ли высок уровень нынешнего развития нейронных технологий, и, в частности, технологий компьютерного зрения, для внедрения их в мусороперерабатывающую отрасль с целью оптимизации всего процесса? Для ответа на поставленный вопрос в работе рассматривается найденная в свободном доступе свёрточная нейронная сеть с открытым исходным кодом, основанная на библиотеке Keras, и возможность её применения на реальном наборе данных. В процессе проведения исследования сравнивается возможность классификации мусора по изображениям из заготовленного для обучения нейросети набора данных и полученного после этого результата со вторым набором данных, подготовленным специально для оценки точности полученного решения после его обучения на первом наборе данных, и, соответственно, полученным после этого вторым результатом.**

Ключевые слова — Компьютерное зрение, Глубокое обучение, Классификация мусора, Распознавание мусора, Kaggle

I. Введение

За последние несколько лет экологичность производства и снижение отходов стало популярным трендом среди крупных фирм и предприятий ввиду массового одобрения такой политики компании у покупателей. Одним из методов работы в данном направлении является классификация и последующая переработка отходов компаний. Из самых известных холдингов и фирм, кто начал следовать данной политике, можно выделить такие мировые бренды как Apple, Coca-Cola, Unilever, Procter & Gamble, сеть американских универмагов Walmart, IKEA.

При создании классификатора мусора важной задачей является распознавание физических признаков отходов, относящих их по созданной автором модели классификации к одному из нескольких возможных типов отходов, и дальнейшая сортировка этого мусора [1]. Для решения данной задачи применяются технологии компьютерного зрения [2, 3].

Классификация мусора включает в себя непосредственно распознавание характерных физических свойств рассматриваемого объекта – его материал, форм-фактор упаковки или изделия, прозрачность, цвет или иные отличительные детали, характерные только для определенного класса [4]. В литературе приводится несколько способов распознания данных свойств, в том числе даже с применением инфракрасных и лазерных сенсоров [5].

Методы глубокого обучения показали высокую производительность и способность к обобщению в задачах данного типа – особенно таких как обнаружение и классификация [6]. В связи с этим, существует множество любительских разработок, представляющих собой детекторы в различных областях – наземное дорожное движение [7], железнодорожный транспорт, летательные аппараты [8], медицина, биология, городская инфраструктура [9], научная деятельность [10] и множество других сфер [11]. Один из подобных детекторов (с открытым исходным кодом), созданный специально для классификации мусора по 2D изображениям, в данной работе будет рассматриваться и анализироваться с целью определения его пригодности к работе в условиях, приближенных к реальным.

Подходы, основанные на обучении, особенно те, которые используют глубокое обучение, требуют больших объёмов аннотированных данных [12]. В настоящее время в свободном доступе есть большое количество зарубежных наборов данных с классифицированными изображениями мусора. Однако следует также заметить, что подходы, основанные на обучении, требуют больших вычислительных мощностей и времени [2, 3].

II. Наборы данных

Для обучения и тестирования рассматриваемой в данной работе нейросети использовались как уже готовые наборы данных из открытых источников, так и локальные, собранные вручную для объективности результатов проведения тестирования. Рассмотрим используемые наборы

A. Kaggle

Обширный набор данных для обучения нейросети классификации мусора был взят с сайта Kaggle. Данный сборник содержит 2D изображения отходов, разделённые на 10 категорий: бумага, картон, пластик, алюминий, жесть, стекло, аэрозольные распылители, батарейки, зажигалки и флуоресцентные лампы. Для всех указанных подвидов отходов представлены различные форм-факторы подобных изделий в разной степени сохранности. В общей сложности данный набор хранит 3000 фотографий мусора.

а  б 

Рис. 1. Примеры образцов мусора из первого заготовленного заранее набора данных для нейросети из каждой выделенной категории

B. Набор данных для проверки работоспособности

Второй набор данных используется непосредственно для проверки рассматриваемой нейросети, полученной в результате обучения на первом наборе данных, к реальной работе. Второй набор не используется в самом процессе обучения, но после на нём проверяется насколько точно полученная модель способна различать новые входные данные. В этом наборе данных изображения собраны с различных источников – фотографии из интернета, экземпляры с других наборов данных, фотографии, сделанные вручную автором статьи. Подобная выборка позволяет оценить, насколько обученная модель приспособлена к работе с реальными отходами, а не только с теми, на которых обучалась [4]. В данном наборе хранится уже около 4000 фотографий различного мусора, классифицированных на те же 10 категорий.

Рис. 2. Примеры образцов мусора из второго проверочного набора данных для нейросети из каждой выделенной категории

III. Нейросетевая архитектура

Свёрточная нейронная сеть в Keras

Для достижения поставленной задачи в данном решении используется свёрточная нейронная сеть в Keras. Сверточная нейронная сеть (на английском языке Convolutional Neural Network или CNN) является распространенной архитектурой глубокого обучения, которая обрабатывает данные с применением операции свертки, когда как Keras предоставляет простой и интуитивно понятный программный интерфейс для создания и обучения сверточных нейронных сетей [2].

Полная архитектура данной нейросети представлена ниже на рисунке 3.

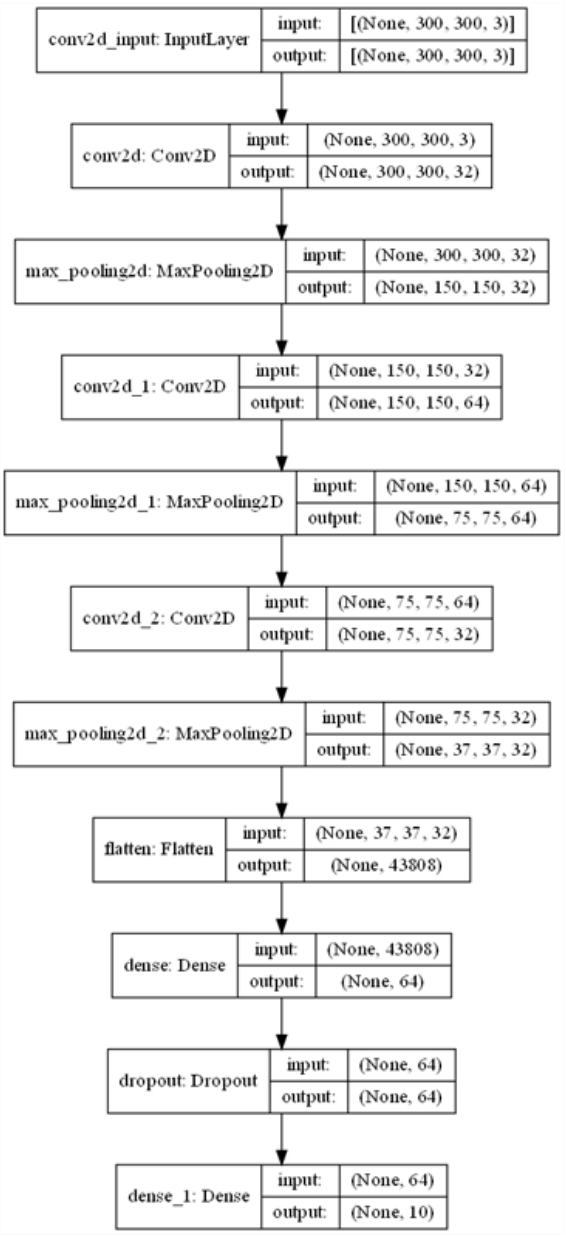


Рис. 3. Архитектура свёрточной нейронной сети

Первый пункт InputLayer с параметрами input: (None, 300, 300, 3) означает, что это начальный слой сверточной нейронной сети в библиотеке Keras [2]. Он представляет входные данные модели.

Аргументы None, 300, 300, 3 указывают на форму входных данных. Каждое измерение имеет следующее значение:

- None — первое измерение обозначает размер батча, то есть количество примеров, которые можно обработать одновременно. Здесь значение "None" означает, что размер батча может быть динамическим и может изменяться в зависимости от входных данных.
- 300 — второе измерение обозначает высоту (высоту изображения) в пикселях.
- 300 — третье измерение обозначает ширину (ширину изображения) в пикселях.
- 3 — четвёртое измерение обозначает количество каналов изображения. Здесь "3" указывает на трехканальное изображение RGB, где каждый канал представляет отдельный цвет (красный, зеленый, синий).

Данный слой ожидает входные данные в формате (батч, высота, ширина, каналы). В контексте модели указание входного слоя InputLayer с этими параметрами определяет форму входных данных для последующих слоев в нейронной сети.

Второй пункт схемы Conv2D Input: (None, 300, 300, 3) означает, что входные даны для этого слоя следующие:

- None — в этом контексте означает пакетное измерение и может принимать любое значение, но обычно это количество образцов (изображений), которое вы помещаете в сеть за один раз.
- 300, 300 — это пространственные измерения изображения, т. е. его высота и ширина.
- 3 — это количество каналов в изображении. Для цветного изображения RGB обычно используется 3 канала (красный, зеленый и синий).

Output: (None, 300, 300, 32) описывает выходные данные с этого слоя.

- None — по-прежнему относится к пакетному измерению.
- 300, 300 — это новые пространственные измерения изображения после прохождения через слой.
- 32 — это количество фильтров, используемых в сверточном слое. То есть модель на этом слое получает 32 различных "версии" входного представления, каждая из которых фокусируется на различных функциях.

Третий пункт MaxPooling2D — это операция которая используется для уменьшения пространственного размера входного представления с сохранением наиболее важной информации. Это делается для уменьшения вычислительной сложности и различной пространственной информации.

Входные данные (None, 300, 300, 32) - это размерность тензора перед применением слоя MaxPooling2D.

Выходные данные (None, 150, 150, 32) - это размерность тензора после применения слоя MaxPooling2D. Обратите внимание, что пространственные размерности (300, 300) уменьшились вдвое до (150, 150), что видно после применения операции MaxPooling с размером пула 2x2, в то время как количество каналов осталось неизменным, т.е. "32". Операция MaxPooling применяется независимо к каждому каналу входного представления.

Пункты 4, 5, 6 и 7 повторяю два предыдущих пункта с постепенным уменьшением размерности.

Пункт 8 — слой Flatten используется для преобразования входных данных, имеющих многомерную структуру, в одномерный вектор. Он "разглаживает" или "сжимает" входные данные, сохраняя при этом информацию о форме входа.

В данном случае, "Flatten input: (None, 37,37,32)" указывает, что входные данные имеют размерность (None, 37, 37, 32), где "None" означает, что батч-размер

может быть произвольным, 37 и 37 - это размеры двумерных входных данных, и 32 - это количество каналов.

"Output: (None, 43808)" указывает, что после применения слоя Flatten, входные данные будут преобразованы в одномерный вектор размерности (None, 43808). Здесь "None" снова означает произвольный батч-размер, а 43808 — это количество элементов в одномерном векторе, полученном из исходных многомерных данных. Количество элементов рассчитывается как произведение размеров всех измерений входных данных после "разглаживания".

Пункт 9 — слой Dense выполняет полносвязную операцию, где каждый нейрон входного тензора соединен с каждым нейроном выходного тензора. В данном случае входной тензор представляет собойвектор размерности (43808,), а выходной тензор - вектор размерности (64,). Это означает, что слой Dense будет иметь 64 нейрона и каждый нейрон будет принимать входные значения от всех 43808 элементов.

Пункт 10 - Dropout является методом регуляризации, который применяется для предотвращения переобучения модели. Он основан на случайном исключении (отключении) заданной доли входных единиц (функций) нейронной сети на каждом обновлении параметров во время обучения. Он помогает избежать переобучения путем случайного исключения некоторых входных единиц на каждом обновлении модели. Это приводит к уменьшению взаимозависимости нейронов и способствует более устойчивому обобщению модели, что часто приводит к улучшению ее общей производительности.

В указанном примере "Dropout input: (None, 64) Output: (None, 64)" означает, что слой Dropout применяется на входной слой с размерностью (None, 64) где 64 обозначает количество входных единиц (нейронов) в слое. Выходной слой Dropout имеет такую же размерность (None, 64), что означает, что количество входных и выходных единиц остается неизменным после применения Dropout.

Пункт 11 повторяет пункт 9 с отличием в том, что данный слой будет иметь итоговые 10 нейронов, которые будут представлять итоговую систему классификации отходов, и каждый нейрон будет принимать входные значения от 64 элементов.

IV. Проведение испытаний

Для тестирования работоспособности выбранной нейронной модели сперва её требуется обучить. В данном случае переменная, настройка которой напрямую повлияет на успешность обучения это выбор количества эпох для тренировки нейросети. Эпоха обозначает один проход через все обучающие примеры в заданном наборе данных. Во время одной эпохи нейронная сеть проходит через все входные данные и обновляет веса своих параметров, чтобы минимизировать ошибку и улучшить свою производительность. Чем больше эпох, тем больше времени потребуется для обучения сети, но при этом повышается шанс достижения лучшей производительности [4, 5]. Для проведения тестирования было решено установить 100 эпох как оптимальное число, при котором время обучения модели будет длиться относительно быстро и при этом с показательным итоговым результатом. Такое число эпох также выбрано в связи с ограниченной вычислительной мощностью оборудования испытателя. Разработчик данной нейронной модели советует ставить как минимум 700 эпох, но подобное обучение занимает по длительности больше нескольких суток, что не является оптимальным временем для показательного тестирования пригодности разработки к эксплуатации.

Результат обучения в течение 100 эпох можно наблюдать на рисунке 4:

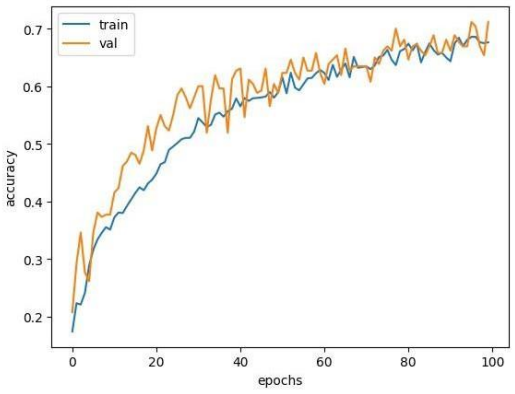


Рис. 4. Результат обучения нейронной модели

По данному графику можно наблюдать, что точность обучения достигла 73%. При этом график не вышел на «плато» и продолжает расти – следовательно, оптимальным количеством эпох для данной разработки является от 170 до 200 эпох. Данное количество позволит сэкономить время на обучении, получить оптимальный результат и избежать переобучения модели [4, 5] - явления, когда модель слишком точно подстраивается под обучающие данные, что приводит к плохой обобщающей способности на новых неизвестных раннее данных. Переобучение возникает, когда модель слишком сложна и слишком точно запоминает особенности обучающего набора данных, вместо обобщения свойств данных.

После получения обученной модели, готовой к обработке подготовленных для неё датасетов, запустим итоговый процесс классификации первого набора данных, на котором и происходил процесс обучения. Точность работы классификатора мусора после проверки работоспособности возросла до 81% по сравнению с 73% из обучения.

Наиболее вероятная причина данного поведения это недообучение [4, 5] - во время обучения модель не смогла полностью запомнить или уловить закономерности в данных. В этом случае модель может показывать низкую точность на обучающих данных, но при этом высокую точность на реальных данных, так как ей удалось улавливать общие закономерности более эффективно. В таком случае подтверждается необходимость в использовании дополнительных эпох.

На рисунке 5 можно наблюдать матрицу ошибок, построенную на основе работы данной модели на исходных данных, где можно рассчитать, что точность нейросети на исходном наборе данных составляет 81%.

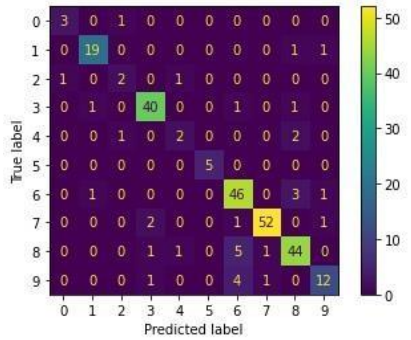


Рис. 5. Матрица ошибок первого запуска

Для второго запуска будет использоваться самостоятельно-подготовленный набор данных, на котором не происходило обучения классификатора мусора, но при этом второй набор содержит изображения, соответствующие заданной в нейросети классификации отходов. Следовательно, данная модель должна распознать и классифицировать эти изображения точно так же, как и во время обучения. Успешным результатом в данном случае будет считаться точность, равная или чуть меньшая, чем точность первого запуска ввиду выявленного явления неодообучения [12] и связанных с этим возможных проблем распознавания новых изображений в хранилище данных.

На рисунке 6 можно наблюдать матрицу ошибок второго запуска. В рамках оценки точности работы классификатора мусора на самостоятельном наборе данных можно наблюдать незначительное снижение с 81% до 79%, что в рамках проведения данного эксперимента является приемлемым результатом и даже успешным.

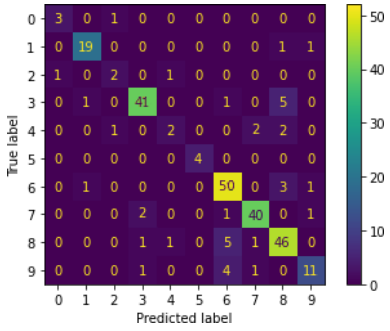


Рис. 6. Матрица ошибок второго запуска

Основываясь на полученных результатах, можно сделать предварительный вывод о том, что обученный на первом наборе данных классификатор мусора успешно справился с поставленной перед ним задачей и готов к работе с реальными данными в лице второго набора [4, 5]. Ввиду ограниченности технологического оборудования при проведении тестирования, данный результат может быть улучшен благодаря большему количеству эпох обучения [5].

V. Заключение

В рамках проведённого исследования было рассмотрено бесплатное готовое решение с открытым исходным кодом - классификатор мусорных отходов, основанный на принципе глубокого обучения с использованием библиотеки Keras [6, 7], и также оценена его готовность к внедрению на реальных мусороперерабатывающих заводах и на линиях сортировки отходов крупных фирм-производителей.

Для тестирования пригодности классификатора мусора к работе в реальных условиях использовались два набора данных – первый набор данных был заранее заготовлен разработчиком, второй же набор данных был собран вручную автором статьи для максимально объективной оценки полученных результатов [6]. Полученные результаты продемонстрировали, что данная модель способна эффективно классифицировать элементы как заготовленных данных, использованных для обучения этой нейросети, так и элементы реальных данных, основанных на той же классификации отходов.

Исходя из всего вышесказанного, можно подвести итог, что классификатор мусора — это перспективная нейронная разработка, рабочие экземпляры которой уже активно создаются как профессионалами, так и энтузиастами. Подобные, даже бесплатные, образчики способны эффективно классифицировать реальные отходы, что свидетельствует о том, что сфера сортировки мусора готова к внедрению нейронных решений и стала на ступень ближе к завершению процесса автоматизации данного процесса и исключению оттуда человеческого труда.

Литература

[1] Аггарвал, Ч. Нейронные сети и глубокое обучение : учебный курс. – М. : Диалектика, 2020. – 744 с. : ил. – ISBN 978-5-907203-01-3.

[2] Джулли, Пал: Библиотека Keras - инструмент глубокого обучения / пер. с англ. А. А. Слинкин.- ДМК Пресс, 2017. – 249 с.

[3] Ян Эрим Солек. Программирование компьютерного зрения на языке Python / пер. с англ. Слинкн А. А. - М.: ДМК Пресс, 2016 - 312 с.: ил.

[4] Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. – СПб. : Питер, 2018. – 480 с. : ил. – ISBN 978-5-496-02536-2.

[5] Рассел С., Норвиг П. Искусственный интеллект: современный подход, 2-е изд.. : Пер. с англ. - М. : Издательский дом ‘‘Вильямс’’, 2007. - 1408 с.

[6] Макшанов, А.В. Технологии интеллектуального анализа данных: Учебное пособие / А.В. Макшанов, А.Е. Журавлев. - СПб.: Лань, 2018. - 212 с.

[7] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[8] Ali, B., Sadekov, R.N. & Tsodokova, V.V. A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. *Gyroscopy Navig.* **13**, 241–252 (2022). https://doi.org/10.1134/S2075108722040022

[9] Chernyshova, Yulia & Savelyev, B & Solodov, S & Pronichkin, S. (2022). Applying distributed ledger technologies in megacities to face anthropogenic burden challenges. IOP Conference Series: Earth and Environmental Science. 1069. 012028. 10.1088/1755-1315/1069/1/012028.

[10] Efimoff, Albert & Matveev, Philip. (2022). Искусственный интеллект для науки и наука для искусственного интеллекта. Voprosy filosofii / Akademiia nauk SSSR, Institut filosofii. 95. 10.21146/0042-8744-2022-

3-93-105.
[11] Полевой, Дмитрий & Kulagin, Petr & Ingacheva, Anastasia & Soldatova, Zhanna & Chukalina, Marina & Nikolaev, Dmitriy & Arlazarov, Vladimir. (2023). From tomographic reconstruction to automatic text recognition: the next frontier task for the artificial intelligence. 51. 10.1117/12.2680132.
[12] В. Ш. Берикашвили, С. П.Оськин Статистическая обработка данных, планирование эксперимента и случайные процессы : учебное пособие для вузов - 2-е изд., испр. и доп. - М. : Юрайт, 2021. - 163 с.

Построение карты пространства вокруг автомобиля на основе видеопоследовательности

А. А. Кожухов
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия m1903540@edu.misis.ru

П. Д. Хонер
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия khonerworki@gmail.com

Аннотация — В данной статье было проведено сравнение двух инновационных подходов в области автономного вождения использующих только одну цветную камеру для построения представления “с высоты птичьего полета” в дорожной сцене и сравнены их результаты на других данных. Первый подход предсказывает расположение дороги и движущихся объектов, учитывая невидимые части мира на изображении с помощью глубокой нейронной сети MonoLayout, которая работает в реальном времени. Данная сеть представляет макет сцены как многоканальную семантическую сетку занятости и использует обучение состязательным функциям, чтобы додумывать правдоподобные дополнения для закрытых частей изображения. Этот подход превосходит текущие методы на различных наборах данных, по словам авторов.

Второй подход фокусируется на использовании модуля трансформации между представлениями учитывая циклическую последовательность между проекциями, а контекстно чувствительный дискриминатор дополнительно улучшает результаты. Эксперименты показывают, что предложенный метод превосходит конкурентов в задачах оценки расположения и загруженности дорог транспортными средствами, обеспечивая высокую производительность в реальном времени.

Ключевые слова — bev, monolayout, cross-view, глубокая нейронная сеть, городская дорожная обстановка, монокулярная камера, беспилотные автомобили, resnet-18, cycledview

I. Введение

С внушительным развитием в области автономного вождения начали появляться увлекательные исследования, связанные с восприятием окружающей среды и анализом сцен [1,2]. В отличие от распространенных решений, которые опираются на дорогостоящие сенсоры lidar и точное глобальное позиционирование, интересным направлением становится расширение возможностей обычных камер [3,4]. В этой (парадигме или контексте) возникает новая и весьма непростая задача построения bev с использованием всего лишь одного цветного изображения. Самая сложная часть этой задачи – это борьба с шумами и искажениями [5].

Две рассмотренные статьи [6,7] вносят значительный вклад в решение данной задачи, предлагая новые методы преобразования изображений.

В первой статье обсуждается глубокая модель MonoLayout, направленная на виртуальное представление динамических и статических объектов дорожной сцены, которое учитывает не только видимые на изображении объекты, но и скрытые/затемнённые. Способность модели восстанавливать атрибуты без прямых визуальных доказательств является важным дополнением к современным методам распознавания и обнаружения объектов.

Вторая статья затрагивает более обширный спектр задач, связанных с автономным вождением, включая оценку композиции сцены, обнаружение 3D-объектов, предсказание поведения транспортных средств и обнаружение полос движения. В ней представлена новая GANоснованная структура, нацеленная на оценку композиции дороги и загруженность дороги транспортными средствами видом сверху.

Обе статьи подчеркивают значимость разработки методов, способных оперировать с минимальным объемом входных данных, расширяют возможности монокулярных видеосенсоров в контексте автономного вождения, строят bev.

II. Наборы данных

Для оценки работы и сравнения двух архитектур сетей были выбраны два набора данных, KITTI [8] и Argoverse [9].

A. KITTI

KITTI – это популярный набор данных для задач компьютерного зрения, предназначенный для исследований в области автономного вождения. Он содержит разнообразные задачи для исследователей, включая обнаружение объектов и понимание сцены. Набор данных получен на основе платформы автономного вождения, разработанной Технологическим институтом Карлсруэ и Технологическим институтом TOYOTA в Чикаго [10].

Набор данных KITTI включает в себя коллекцию различных датчиков и модальностей, таких как стереокамеры, лидары и датчики GPS, что позволяет получить полное представление о среде вокруг автомобиля. Данные собирались в течение нескольких дней в городских районах Карлсруэ и близлежащих городах Германии. Набор данных включает в себя более 200 000 стереоизображений. Пример изображения из рассматриваемого набора данных представлен на рисунке 1.

Рисунок 1 – Пример изображения из датасета KITTI

Набор данных разделен на несколько различных категорий, каждая из которых имеет свой собственный набор задач. К этим категориям относятся обнаружение объектов, отслеживание, понимание сцены, визуальная одометрия и обнаружение дорог и дорожных полос. В данной работе будет использоваться датасет KITTI Odometry.

B. Argoverse

Набор данных Argoverse – это коллекция данных, предназначенных для поддержки исследований в области задач автономного вождения, таких как понимание сцены вокруг автомобиля, 3D-слежение и прогнозирование движения. Разработанный компанией Argo AI [11], набор данных содержит широкий спектр высококачественных сенсорных данных, включая изображения высокого разрешения и картографические данные. Этот набор данных содержит более 290 000 маркированных объектов и 5 млн экземпляров объектов на 1 263 различных сценах. Датасет состоит из трех основных поднаборов, данной статье будет использоваться один из них. Argoverse Tracking – это коллекция из 113 логов сегментов с аннотациями по отслеживанию объектов. Эти логи сегментов, называемыми “последовательностями”, имеют длину от 15 до 30 секунд и содержат в общей сложности 11 052 последовательности.

Каждая последовательность в обучающих и валидационных данных включает аннотации для всех объектов в радиусе 5 метров от того, что определяется как “область, пригодная для движения” – область, в которой возможно движение автомобиля. Пример изображения из датасета Argoverse представлен на рисунке 2.

Рисунок 2 – Пример изображения из датасета Argoverse

III. Нейросетевые архитектуры

A. MonoLayout

1. Архитектура сети

Данная сеть рассматривает проблему амодальной оценки расположения сцены по одному цветному изображению. Формально, учитывая цветное изображение, полученное с камеры, мы стремимся предсказать расположение статических и динамических элементов с высоты птичьего полета. В частности, оцениваются следующие три величины: 1) набор всех статических точек сцены S (обычно дорога и тротуар) на плоскости земли вне зависимости от того, изображены они на изображении или нет; 2) набор всех динамических точек сцены D на плоскости земли, занятых автомобилями, все зависимости от того, изображены они на изображении или нет. 3) для каждой точки, определенной в пунктах 1 и 2, и которая относится к транспортным средствам, привязывается маркировка конкретного экземпляра, к какому типу транспортного средства относится данная точка.

Архитектура MonoLayout, представленная на рисунке 3, состоит из четырех подсетей:

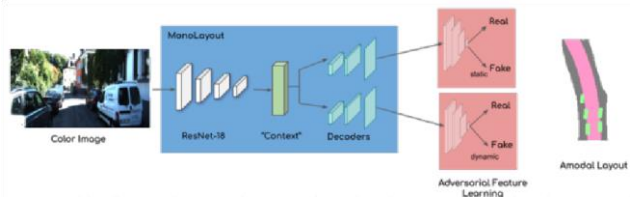


Рисунок 3 – Архитектура: Сеть MonoLayout получает цветное изображения сцены городского вождения и предсказывает амодальную схему сцены с высоты птичьего полета. Архитектура включает в себя кодировщик контекста, декодеры амодального расположения и два дискриминатора

- Из входного изображения сначала извлекаются значимые признаки с помощью ResNet-18, дообученного для определения статических и динамических аспектов сцены;
- Далее контекстный энкодер извлекает многомерные представления признаков из полученного вектора ResNet кодировщика. Это обеспечивает общий контекст, который захватывает статические и динамические компоненты сцены для последующей обработки;
- Далее следует амодальный декодер статических сцен, который декодирует общий контекст для получения амодального макета статической сцены. Эта модель состоит из серии сверточных слоев и апсемплинга, которые отображают общий контекст на статическую сцену с высоты птичьего полета;
- Далее данные переходят в динамический декодер сцена, архитектурно схожий со статическим декодером, который предсказывает местонахождение автомобилей с высоты птичьего полета;
- В результате данные помещаются в два дискриминатора, которые регуляризуют предсказанные статичееские и динамические схемы расположения путем регуляризации их распределений, чтобы они были похожи на истинные распределения правдоподобных геометрий дорог и истинного числа транспортных средств.

Статический и динамический декодер имеют идентичную архитектуру. Они декодируют общий контекст из результатов работы ResNet с помощью серии слоев апсемплинга.

2. Функция ошибок

Параметры ϕ, ϑ, ψ контекстного энкодера, статического и динамического декодера получаются путем их минимизации с помощью стохастического градиентного спуска по батчам.

$$\min_{\nu,\psi,\theta_S,\theta_D} \mathcal{L}_{\text{норм}}(\phi,\nu,\psi) + \mathcal{L}_{\text{сост}}(\phi,\theta,\psi) + \mathcal{L}_{\text{дискрим}}(\phi,\nu)$$

$$\mathcal{L}_{\text{норм}} = \sum^{N} \left|\left|\mathcal{S}_{\phi,\nu}(\mathcal{I}^i) - \mathcal{S}^i_{gt}\right|\right|^2 + \left|\left|\mathcal{D}_{\phi,\psi}(\mathcal{I}^i) - \mathcal{D}^i_{gt}\right|\right|^2$$

/45

$$\mathcal{L}_{\text{сост}}(S,D;\phi,\theta,\psi) = \mathbb{E}_{\%\sim 7\#\$\%\&}[(D(\theta_8)-1)_2] + \mathbb{E}_{\%\sim 7\#\$\%\&}[(D(\theta_9)-1)_2]$$

$$\mathcal{L}_{\text{дискрим}}(D;\theta) = 0 \quad \mathbb{E}_{\%\sim 7'()\&}[(D(\theta)-1)_2] \quad \theta\in\theta_D,\theta_S + \mathbb{E}_{\theta\sim p_{fake}}[(D(\theta)-1)^2]$$

Здесь, $\mathcal{L}_{\text{норм}}$ – вторая норма (L2), которая корректирует отклонение предсказанных статических и динамических слоев ($\mathcal{S}_{!,\#}(\mathcal{I})$, $\mathcal{D}_{!,\#}(\mathcal{I})$) от их соответствующих истинных значений ($\mathcal{S}_{01}/$, $\mathcal{D}_{01}/$) . Состязательная ошибка $\mathcal{L}_{\text{сост}}$ способствует тому, чтобы распределение оценок, полученных декодерами статическим и динамических сцен были близкими к истинным аналогам. Ошибка дискриминатора $\mathcal{L}_{\text{дискрим}}$ является целью обновления дискриминатора.

B. Cross-View

1. Архитектура сети

Архитектура сети, представленная на рисунке 4, основана на генеративно-состязательных сетях (GAN). В частности, генератор, представляющий из себя автоэнкодер, в котором входное фронтальное изображение сначала проходит через кодировщик, который использует ResNet в качестве основной сети для извлечения визуальных признаков, затем предлагаемый нами модуль преобразования cross-view, который улучшает характеристики для проекции представления, и, наконец, декодер для получения изображения представления с птичьего полета. С другой стороны, предлагается контекстно-зависимый дискриминатор, представленный на рисунке 6, который различает маски транспортных средств.

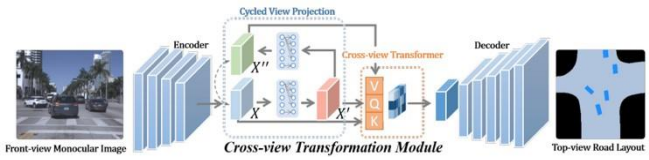


Рисунок 4 – Архитектура сети Cross-View

2. Модуль преобразования cross-view

Из-за большого разрыва между фронтальным изображением и изображения представления сверху, существует большое количество недостающего содержимого изображения при проецировании, поэтому традиционные методы проецирования представления приводят к некачественным результатам. Сверточные нейронные сети позволяют решить данную проблему, но корреляция обоих представлений не является тривиальной для моделирования в глубоких сетях.

Чтобы усилить корреляцию представлений, используя возможности глубоких сетей, вводится модуль преобразования cross-view в генератор GAN архитектуры, который усиливает извлеченные визуальные признаки для проецирования фронтального изображения в изображения вида с птичьего полета.

Структура предложенного модуля преобразования cross-view представлена на рисунке 4, которая состоит из двух частей: циклической проекцией представления и cross-view трансформера.

3. Циклическая проекция представления

Поскольку признаки фронтального представления пространственно не совпадают с признаками представлений сверху из-за их большого разрыва, следую практике, мы используем структуру многослойного перцептрона состоящую из двух полностью связанных слоев, для проецирования признаков фронтальных представлений на представления с птичьего полета, что позволяет обойти стандартный поток информации при суммировании сверточных слоев. Как показано на рисунке с архитектурой, X и X' представляют собой карты признаков до и после проецирования представления соответственно. Таким образом, целостная проекция вида достигается применением многослойного перцептрона на признаки, полученные после применения ResNet.

Однако такая простая структура проекции представления не может гарантировать эффективную передачу информации о фронтальных представлениях. Здесь мы представляем циклическую схему самообучения для консолидации проекции представления, которая проецирует особенности BEV представления обратно в область фронтального представления. Как показано на рисунке 4, X" вычисляется путем циклического возвращения X' обратно в область фронтальных видов с помощью той же структуры многослойного перцептрона. Чтобы гарантировать согласованность между X' и X", мы включаем потерю цикла, представляющую из себя первую норму между X и X".

4. Cross-view трансформер

Основная цель Cross-view трансформера – соотнести признаки до проецирования представления X и признаки после проецирования представления X' для того, чтобы усилить последние. Поскольку X" содержит существенную информацию о фронтальном представлении для проекции представления, она может быть использована для дальнейшего улучшения

характеристик. Как показано на рисунке 5, cross-view трансформер можно условно разделить на две схемы: схема межракурсной корреляции, которая явно коррелирует особенности представлении для получения карты внимания W для усиления X', а также схема выбора признаков, которая извлекает наиболее значимую информацию из. X".

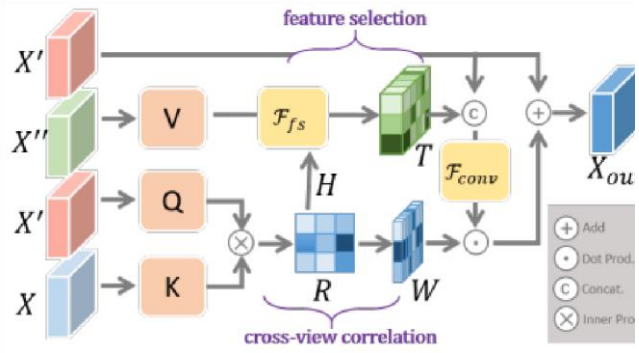


Рисунок 5 – Архитектура cross-view трансформера

IV. Сравнение

Проведем визуальное сравнение работ двух сетей. Результаты представлены на рисунках 7,8, 9 и 10.

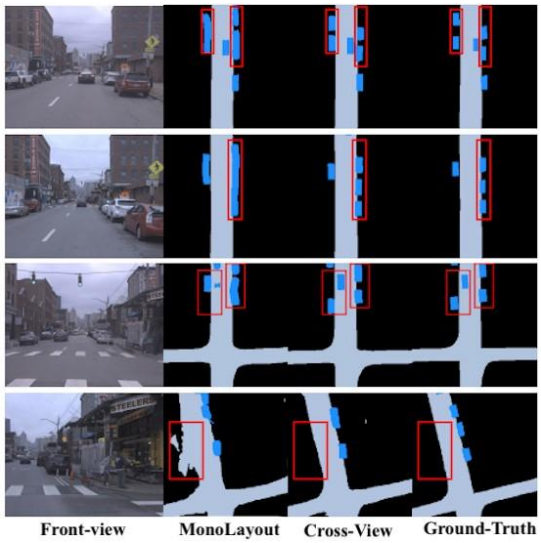


Рисунок 7 – Результаты работы сетей на наборе данных Argoverse (Static + Dynamic)

5. Контекстно-зависимый дискриминатор

В дискриминаторе на основе GAN архитектуры для дальнейшего уточнения синтетических масок автомобилей можно использовать пространственную связь между автомобилями и их контекстом (то есть дорогой). Для этого предлагается контекстный дискриминатор, который не только пытается отличить маски транспортных средств, но и явно использует корреляцию между транспортными средствами и дорогами для усиления влияния.
Архитектура дискриминатора представлена на рисунке 6.

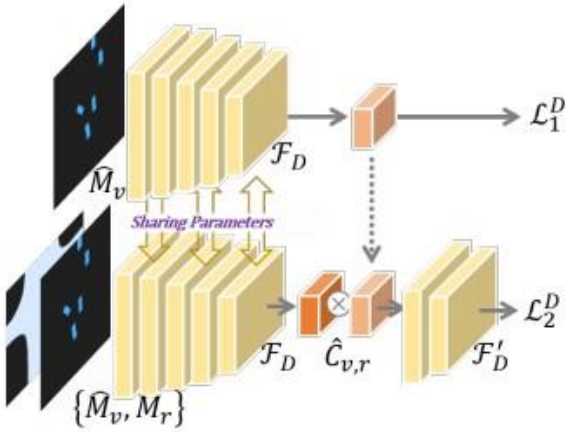


Рисунок 6 – Архитектура контексто-независимого дискриминатора

6. Функция ошибок

Функция потерь в данной системе определяется как сумма функции потерь бинарной кросс-энтропии, балансовой циклической функции потерь и коэффициента и суммы состязательных потерь на коэффициент. На практике, циклический и состязательный коэффициенты равны 0,001 и 1 соответственно.

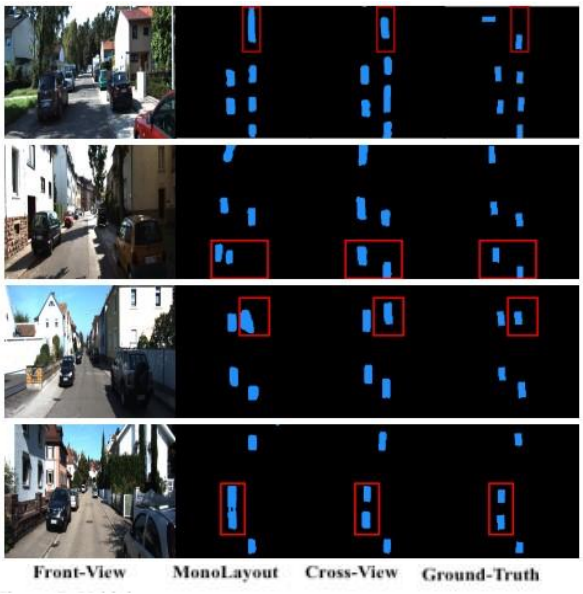


Рисунок 8 – Результаты работы сетей на наборе данных KITTI 3DObject

Рисунок 9 – Результаты работы сетей на наборе данных KITTI Odometry

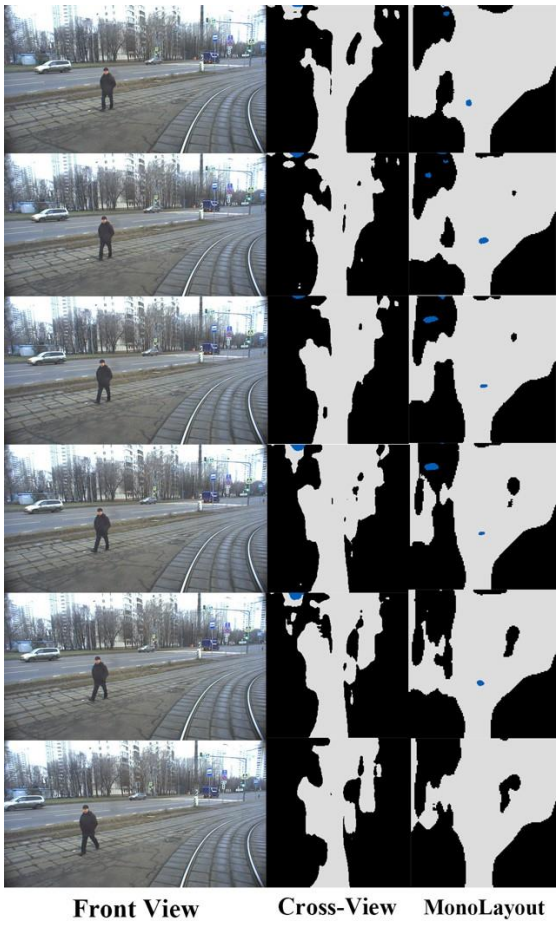


Рисунок 10 – Результаты построения карты пространства двух сетей

При визуальном анализе видно, что сеть Cross-View более четко отрисовывает пространство в виде с птичьего полета. Можно сказать, что при анализе сегментации транспортных средств вокруг сеть Cross-View лучше, но ненамного. При построении карты пространства вокруг трамвая видно, что сеть работает не совсем корректно: Предположительно, это связано с тем, что сети обучались исключительно на видео с автодорог, при обучении на датасете с движениями трамваев изображение будет более точным.

Рассмотрим численные результаты тестирования сетей:

Таблица 1. Результаты работы сетей на наборе данных KITTI 3DObject (Vehicles)

Наименование сети	mIOU (%)	mAP (%)
MonoLayout	30.18	45,91
Cross-View	38,85	51,04

Таблица 2. Результаты работы сетей на наборе данных KITTI Odometry (road)

Наименование сети	mIOU (%)	mAP (%)

MonoLayout	76,15	85,25
Cross-View	77,47	86,39

Таблица 3. Результаты работы сетей на наборе данных Argoverse (Dynamic)

Наименование сети	mIOU (%)	mAP (%)
MonoLayout	32,58	51,06
Cross-View	47,87	62,69

Таблица 4. Результаты работы сетей на наборе данных Argoverse (Static)

Наименование сети	mIOU (%)	mAP (%)
MonoLayout	73,25	84,56
Cross-View	76,56	87,3

По полученным результатам можно сделать вывод, что по всем показателям Cross-View работает лучше

V. Заключение

Были рассмотрены наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети. А также проведены исследования на новых данных. Рассмотрены два нестандартных подхода к решению задач с построением BEV, оценки городской дорожной обстановки, восстановлении HD-карт. Каждая сеть рассмотрена с точки зрения её уникальной архитектуры и процесса обучения.

Приведённые подходы были сравнены на части датасета KITTI, Argoverse и на нашем видео с трамвая. Отдельно было оценено качество локализации и классификации машин в BEV. По полученным данным очевидно, что нейронная сеть Cross-View, подготовленная авторами работы [2], имеет по всем показателям преимущество перед альтернативным подходом, что объясняется использованием Cycled-view трансформера в архитектуре. Притом сравнивались именно конкретные модели с конкретными весами. Стоит сказать, что обе сети показали себя хорошо и в будущем послужат основой для других сетей.

Литература

[1] Berkovich, S.B. *et al.* (2017) ‘Using inertial navigation systems to monitor the motion of a train’, *2017 24th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins.2017.7995623.

[2] Bikmaev, R.R. *et al.* (2019) ‘Improving the accuracy of supporting mobile objects with the use of the algorithm of complex processing of signals with a monocular camera and Lidar’, *2019 26th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins.2019.8769360.

[3] Efimov, A.R. (2022) ‘AI for Science and Science for AI’, *Voprosy Filosofii*, pp. 93–105. doi:10.21146/0042-8744-2022-3-93-105.

[4] Kotov, N.I. *et al.* (2017) ‘Using Vision Systems to determine the vehicle position on the road’, *2017 24th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins.2017.7995567.

[5] Sadekov, R.N. *et al.* (2017) ‘Road sign detection and recognition in panoramic images to generate navigational maps’, *2017 24th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins.2017.7995611.

[6] K. Mani, S. Daga, S. Garg, S. S. Narasimhan, M. Krishna, and K. M. Jatavallabhula, “MonoLayout: Amodal scene layout from a single image,” in *The IEEE Winter Conference on Applications of Computer Vision*, 2020, pp. 1689–1697.

[7] Y. Weixiang, L. Qi, L. Wenxi, Y. Yuanlong, M. Yuexin, H. Shengfeng, P. Jia, “Perojecting Your View AttentivelyL Monocular Road Scene Layout Estimation vie Cross-view Transformation” in CVPR, 2021, pp/ 15536–15545.

[8] A. Geiger, P. Lenz, and R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite,” 2012.

[9] M.-F. Chang et al., “Argoverse: 3D Tracking and Forecasting with Rich Maps,” 2019

[10] KITTI Dataset. URL – https://armanasq.github.io/datsets/kitti/.

[11] M.-F. Chang et al., “Argoverse: 3D Tracking and Forecasting with Rich Maps,” in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 8748–8757

Классификация драгоценных камней при помощи компьютерного зрения

А. А. Фомина
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2300443@edu.misis.ru

***Аннотация*— Драгоценные камни представляют собой изысканные материалы, обладающие высокой ценностью. В настоящей работе рассматриваются различные решения с использованием нейронных сетей, а также проводится сравнение их возможностей по классификации драгоценных камней при помощи компьютерного зрения, используя различные наборы данных. Цель данного исследования является выявление эффективных моделей, способных качественно классифицировать драгоценные камни. Анализ проведенных исследований не только предоставляет обзор на различные методы классификации драгоценных камней, но и подчеркивает сложности в поиске эффективных решений. Это исследование имеет высокую значимость в контексте развития современных методов геологической диагностики и обогащения научных знаний в области минералогии.**

Ключевые слова — драгоценные камни, геммология, компьютерное зрение, аугментация данных, нейронная сеть, CNN, Inseption V3, Resnet50, Gemstones Images

I. Введение

Аккуратная классификация драгоценных камней представляет собой важный этап в оценке и экспертизе этих ценных материалов, поскольку их идентификация является ключевым моментом в данном процессе. В настоящее время идентификация драгоценных камней осуществляется с использованием визуального наблюдения и спектрохимического анализа. Геммологи [1], внимательно исследуя камни невооруженным глазом и под увеличением, выявляют визуальные характеристики, такие как цвет, прозрачность, блеск, трещины, крошения, включения, плеохроизм, явления, двойное лучепреломление для разделения драгоценных камней и другие характеристики. Этот процесс сложен, поскольку многие камни могут иметь схожие цвета и характеристики. Важно отметить, что идентификация и классификация часто сопровождаются использованием геммологических инструментов [2], включая рефрактометры, полярископы и коноскопы, ручные спектроскопы, дихроскопы и ультрафиолетовый свет, чтобы изучать оптические свойства драгоценных камней.

В контексте современных методов геологической диагностики и обогащения научных знаний в области минералогии существует актуальная потребность в разработке автоматических, эффективных и точных методов идентификации драгоценных камней. В последние годы глубокое обучение [3], представленное сверточными нейронными сетями (CNN) [4], получило широкое применение в области классификации изображений [5, 6] и показало впечатляющие результаты. Это вдохновило нас на исследование возможности применения таких сетей для идентификации драгоценных камней, анализируя их визуальные характеристики.

В свете этих обстоятельств, в данной работе представляется методика компьютерного зрения [7, 8, 9] для автоматизированной классификации 87 категорий драгоценных камней. В работе рассматриваются и сравниваются достижения 2 сверточных нейронных сетей: Inseption V3 [10], и Resnet50 [11, 12], (с открытым исходным кодом) в области глубокого обучения для классификации драгоценных камней.

II. Наборы данных

В данной работе представлены 2 набора данных, как взятых из открытых источников, так и самостоятельно собранных. Рассмотрим используемые наборы.

A. Gemstones Images[13]

Обширный набор общедоступных данных Gemstones Images, содержащий более 3200 изображений различных драгоценных камней (рисунок 1).

Рис. 1. Примеры драгоценных камней из 87 различных классов

На рисунке изображения камней расположены по оттенкам, чтобы проиллюстрировать сложность классификации драгоценных камней при визуальном осмотре. Набор данных содержит 87 классов (рисунок 2). Каждый класс представляет собой отдельный вид драгоценных камней.

Alexandrite	Chrysocolla	Larimar	'Sapphire Blue'
Almandine	Chrysoprase	Malachite	'Sapphire Pink'
Amazonite	Citrine	Moonstone	'Sapphire Purple'
Amber	Coral	Morganite	'Sapphire Yellow'
Amethyst	Danburite	'Onyx Black'	Scapolite
Ametrine	Diamond	'Onyx Green'	Serpentine
Andalusite	Diaspore	'Onyx Red'	Sodalite
Andradite	Dumortierite	Opal	Spessartite
Aquamarine	Emerald	Pearl	Sphene
Aventurine	Fluorite	Peridot	Spinel
'Aventurine Green'	'Garnet Red'	Prehnite	Spodumene
Benitoite	Goshenite	Pyrite	Sunstone
'Beryl Golden'	Grossular	Pyrope	Tanzanite
'Beryl Red'	Hessonite	'Quartz Beer'	'Tigers Eye'
Bloodstone	Hiddenite	'Quartz Lemon'	Topaz
'Blue Lace Agate'	Iolite	'Quartz Rose'	Tourmaline
Carnelian	Jade	'Quartz Rutilated'	Tsavorite
'Cats Eye'	Jasper	'Quartz Smoky'	Turquoise
Chalcedony	Kunzite	Rhodochrosite	Variscite
'Chalcedony Blue'	Kyanite	Rhodolite	Zircon
'Chrome Diopside'	Labradorite	Rhodonite	Zoisite
Chrysoberyl	'Lapis Lazuli'	Ruby	

Рис. 2. 87 классов драгоценных камней

Каждый драгоценный камень обладает своими уникальными физическими и оптическими свойствами, которые отражаются на их внешнем виде. Каждый камень, из 87 классов представлен разным количеством картинок и форм, огранки. Некоторые виды камней крайне схожи с другим видом и это, в значительной мере, затрудняет классификацию. На примере 2 видов камней «Almandine» и «Garnet Red» видно насколько разные классы драгоценных камней могут быть похожи внешне, но на самом деле быть в разных классах (рисунок 3).

Рис. 3. Драгоценные камни схожие внешне: а) almandine, б) garnet red

Также, следует отметить, что существуют некотрые особенности в каждом классе. Рассмотрим на примере класса «Amazonite»:

- форма камней в каждом классе существенно отличается (а, б,);
- количество камней на картинке может быть различным (в, г);
- цвет общего фона может отличаться (д, е);
- в кадр могут попасть посторонние предметы (ж).

Рис. 4. Особенности : а) форма капли, б)не стандартная форма, в) 3 камня на картинке, г) 2 камня на картинке, д) белый фон изображения е) цветной фон изображения, ж) посторонний предмет в кадре

B. Дополненный набор данных

Данный набор данных является дополненным, по отношению к набору данных «Gemstones Images». Дополненный более 150 различными изображениями драгоценных камней, найденных в открытых источниках. Набор данных «Gemstones Images» сам по себе является обширным набором. Дополнительные изображения необходимы для повышения качества проверки нейронных сетей. Также как и в первоначальном наборе, было принято решение дополнить не только стандартными изображениями драгоценных камней, но и соблюсти все частности, например: форма камней, количество камней на изображении, цвет фона и наличие посторонних предметов на рисунке. Примеры дополнительных изображений отражены на рисунке 5.

Рис. 5. Дополнительный набор данных а) форма капли, б) не стандартная форма, в) 2 камня на картинке, г) 3 камня на картинке, д) белый фон изображения е) цветной фон изображения, ж, з) посторонний предмет в кадре

III. Нейросетевые архитектуры

В представленной работе для решения задачи классификации драгоценных камней использовались две нейронные сети: InceptionV3 и ResNet50. Обе модели представляют собой мощные инструменты для извлечения высокоуровневых признаков [14] из изображений, что важно для точной классификации сложных объектов, таких как драгоценные камни.

A. Inseption V3

Для классификации 87 типов драгоценных камней была использована сверточная нейросетевая модель. Модель включает в себя несколько сверточных слоев [15], за которыми следуют объединяющие слои и полностью связанные слои. Inceptionv3 — это одна из глубоких сверточных нейронных сетей, разработанная Google для обработки изображений. Она использует инновационные "Inception" блоки, которые позволяют эффективно извлекать различные уровни признаков из входных изображений. Архитектура [16] модели представлена на рисунке 6.

Для классификации используются веса предобученной на наборе данных ImageNet [17] модели, доступные в TensorFlow. Модель дообучается на изображениях из набора данных Gemstones Images и на самостоятельно собранном наборе данных. Дообучение длится 5 эпох с размером батча 128. Выходной слой модели представляет собой полностью связанный слой с 87 выходными единицами, соответствующими каждому классу драгоценных камней. В качестве оптимизатора был выбран оптимизатор Adam с фиксированной скоростью обучения 0,0001. Для предотвращения переобучения применялась регуляризация весов с использованием L2-нормы. В качестве функции потерь используется кросс-энтропия в задаче мулькласовой классификации (cate-gorical cross-entropy) – softmax активация в сочетании с кросс-энтропийной функцией потерь. Кросс-энтропия (Cross-entropy) [18],

$$C = -\left(\frac{1}{N}\right) * \sum\nolimits_{-}i\,(y\_i * \log\,(y`\_i))$$

где N — количество выборок, y_i — истинная метка для i-й выборки, а y’_i — прогнозируемая вероятность для i-й выборки.

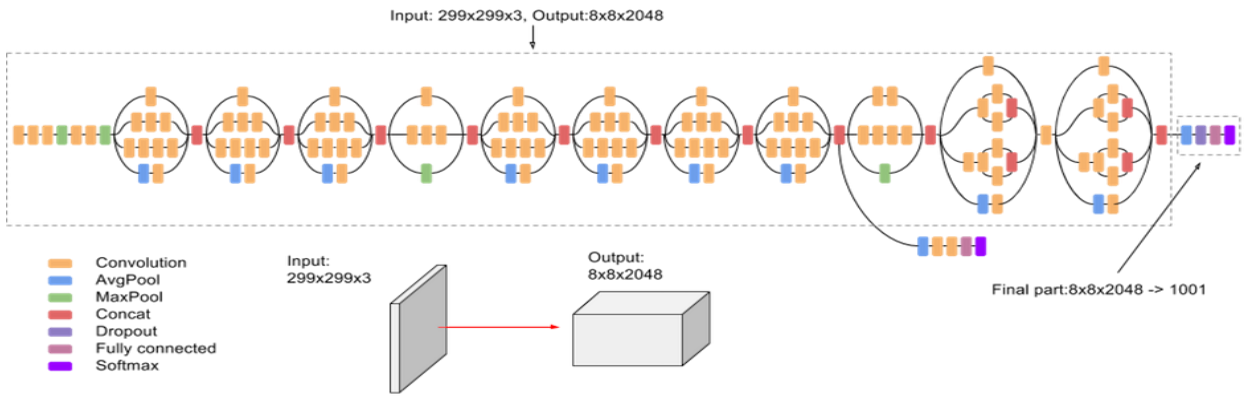


Рис. 6. Архитектура InceptionV3

Также в процесс были включен нормализация и маштабирование, а также Label Encode для кодирования строковых меток классов в числовые индексы, а также аугментация изображений включающую в себя:

- Цветовая коррекция;
- Отражение по горизонтали;
- Случайное вращение.

B. *Resnet50*

ResNet-50 - глубокая сверточная нейросеть, использующая блоки с пропуском (skip connections) для обучения глубоких представлений изображений, содержащая 50 слоев с весами, способствуя эффективной передаче градиентов и борьбе с проблемой затухающих градиентов. Архитектура [19] модели представлена на рисунке 8.

Также, как и у модели Inseption V3 модель дообучалась на собственных двух датасетах. Дообучение длится 5 эпох с размером батча 128. Выход сети это один из 87 классов драгоценных камней, соответствующий классам камней из представленных датасетов. Оптимизатор – Adam, с фиксированной скоростью обучения 0,0001. Применялась регуляризация весов с использованием L2-нормы. Функция потерь - кросс- (cate-gorical cross-entropy) – softmax активация в сочетании с кросс-энтропийной функцией потерь. Все также, как и у предыдущей модели была нормализация и маштабирование, а также Label Encode и аугментация. Аугментация была применена для еще большего расширения обучающего набора, а также повышения устойчивости к различным условиям реального мира, например изменения освещения, углы съемки и т. д. Примеры изображений после аугментации представлены на рисунке 7.

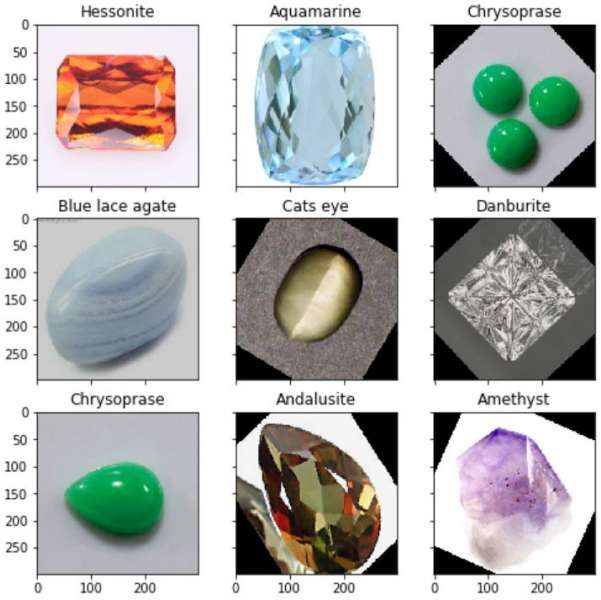


Рис. 7. Примеры изображений после аугментации

IV. Результаты

Точность модели является показателем того, насколько хорошо модель способна правильно классифицировать драгоценные камни. Следует отметить, что все модели показали хорошие результаты. Функция оценок производилась по F1-score. Формула отображена ниже:

$$F1 = 2\frac{Precision \cdot Recall}{Precision + Recall} = \frac{2TP}{2TP + FP + FN},$$

где Precision – точность. Recall – полнота.

В свою очередь:

- Precision (точность) рассчитывается как:

$$Precision = \frac{TP}{TP + FP}.$$

- Recall (полнота) рассчитывается по формуле:

$$Recall = \frac{TP}{TP + FN}.$$

Разберём обозначения TP, FP, FN:

В контексте нашей задачи по анализу драгоценных камней:

- True Positive (TP): это количество драгоценных камней, которые на самом деле присутствуют в выборке, и модель корректно идентифицировала их как драгоценные.
- False Positive (FP): это количество объектов, которые на самом деле являются менее драгоценными камнями, но модель ошибочно предсказала их как другой класс.
- False Negative (FN): это количество драгоценных камней, которые на самом деле присутствуют в выборке, но модель не смогла их идентифицировать.

F1_score для каждой модели отображена в таблице:

Таблица 1. F1_score каждой модели

Модель	**F1_score**
ResNet50	0.5047619047619047
Inseption V3	0.8117647058823529

Из таблицы видно, лучшие результаты имеет модель Inseption V3. Точность модели составляет 0.8117, что означает, что 81,11% положительных прогнозов, сделанных моделью, были правильными. Это хороший признак того, что модель хорошо работает и показывает значительные результаты. Модель ResNet50 показала результаты значительно хуже по сравнению с моделью Inseption V3.

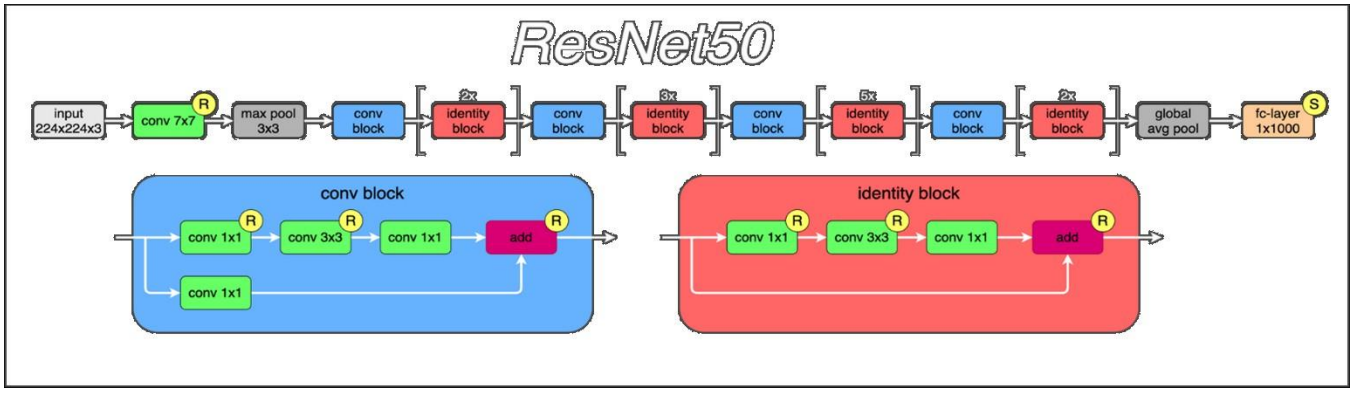


Рис. 8. Архитектура ResNet50

Сделаем визуализацию, чтобы наглядней видеть результаты обучения модели. На рисунке ниже отражены некоторые виды драгоценных камней и процент их правильного определения моделью Inseption V3 (рисунок 9).

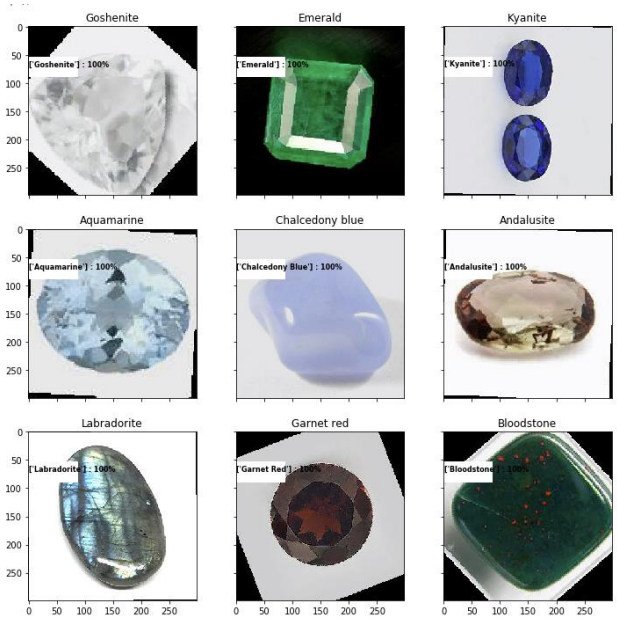


Рис. 9. Процент правильного определения

Следует также обратить внимание на точность для каждого класса драгоценных камней. В связи с тем, что классов крайне много – 87, было принято решение, для улучшения визуализации, отобразить точность в виде таблицы для каждого класса.

Таблица 2 демонстрирует точность модели Inseption V3 по каждому классу драгоценных камней.

Таблица 2. Точность модели Inseption V3 по каждому классу

№	Класс камня	Точность в %	№	Класс камня	Точность в %
1	Alexandrite	100	45	Larimar	95
2	Almandine	65	46	Malachite	95
3	Amazonite	76	47	Moonstone	90
4	Amber	100	48	Morganite	80
5	Amethyst	74	49	Onyx Black	100
6	Ametrine	100	50	Onyx Green	95
7	Andalusite	94	51	Onyx Red	83
8	Andradite	75	52	Opal	100
9	Aquamarine	70	53	Pearl	100
10	Aventurine	95	54	Peridot	89
11	Aventurine Green	88	55	Prehnite	95
12	Benitoite	87	56	Pyrite	100
13	Beryl Golden	92	57	Pyrope	80
14	Beryl Red	86	58	Quartz Beer	100
15	Bloodstone	100	59	Quartz Lemon	95
16	Blue Lace Agate	100	60	Quartz Rose	93
17	Carnelian	91	61	Quartz Rutilated	100
18	Cats Eye	100	62	Quartz Smoky	100
19	Chalcedony	86	63	Rhodochrosite	85
20	Chalcedony Blue	95	64	Rhodolite	80
21	Chrome Diopside	89	65	Rhodonite	86
22	Chrysoberyl	80	66	Ruby	90
23	Chrysocolla	94	67	Sapphire Blue	53
24	Chrysoprase	96	68	Sapphire Pink	75
25	Citrine	70	69	Sapphire Purple	80
26	Coral	90	70	Sapphire Yellow	95
27	Danburite	90	71	Scapolite	100
28	Diamond	78	72	Serpentine	95
29	Diaspore	89	73	Sodalite	84
30	Dumortierite	95	74	Spessartite	86
31	Emerald	87	75	Sphene	90
32	Fluorite	95	76	Spinel	95
33	Garnet Red	55	77	Spodumene	76
34	Goshenite	100	78	Sunstone	100

35	Grossular	70	**79**	Tanzanite	90
36	Hessonite	82	**80**	Tigers Eye	100
37	Hiddenite	65	**81**	Topaz	100
38	Iolite	100	**82**	Tourmaline	100
39	Jade	100	**83**	Tsavorite	65
40	Jasper	95	**84**	Turquoise	100
41	Kunzite	71	**85**	Variscite	100
42	Kyanite	100	**86**	Zircon	58
43	Labradorite	100	**87**	Zoisite	76
44	Lapis Lazuli	100			

Из представленной таблицы видно, что наименее привлекательными результатами обладают камни со схожими визуальными характеристиками. Выше был приведен пример подобных изображений в рисунке 3 на примере классов "Almandine" и "Garnet Red". Класс "Almandine" демонстрирует точность в 65%, в то время как класс драгоценного камня "Garnet Red" достигает лишь 55%. Обнаружено, что эти два класса камней представляют собой особую сложность для классификации. В то время как большинство других классов проявляют значительную точность, приближенную к 100%. Этот результат представляет собой весьма впечатляющее достижение, поскольку модель проявляет высокую точность.

V. Заключение

В настоящем исследовании были подробно рассмотрены основные наборы данных, на которых проводилось обучение и тестирование рассматриваемых нейронных сетей. Кроме того, для эффективной обработки информации был создан собственный набор данных, обеспечивающий более глубокий и точный анализ классификации драгоценных камней.

В работе представлены две различные нейронные сети, применяемых для решения задачи классификации. Каждая нейронная сеть рассмотрена в контексте ее архитектуры, процесса обучения, а также использованных для обучения и тестирования наборов данных. Это позволяет читателю получить полное представление о методологии и технических аспектах проведенного исследования.

Каждая из представленных нейронных сетей была подробно проанализирована, а полученные результаты были подвергнуты сравнительному анализу. По полученным данным можно утверждать, что нейронная сеть InceptionV3 демонстрирует определенные преимущества по сравнению с альтернативной сетью ResNet50. Это выражается в более высокой точности в решении задачи классификации драгоценных камней.

В целом, результаты исследования подчеркивают не только значимость использования современных нейронных сетей в области распознавания драгоценных камней, но и важность выбора оптимальной архитектуры для конкретной задачи.

Литература

[1] Anderson, B.W. (1976) Gemstones for Everyman, Faber and Faber, London

[2] Günther, B. (1981) Tables of Gemstone Identification, Elizabeth Lenzen, Kirschweiler

[3] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[4] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[5] Ali, B., Sadekov, R.N., & Tsodokova, V.V. (2022). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation, 13, 241-252.

[6] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[7] Chernyshova, Yulia, Savelyev, B, Solodov, S Pronichkin, S. (2022). Applying distributed ledger technologies in megacities to face anthropogenic burden challenges. IOP Conference Series: Earth and Environmental Science. 1069. 012028. 10.1088/1755 1315/1069/1/012028.

[8] Efimoff, Albert & Matveev, Philip. (2022). Искусственный интеллект для науки и наука для искусственного интеллекта. Voprosy filosofii / Akademiia nauk SSSR, Institut filosofii. 95.10.21146/0042-8744-2022-3-93-105.

[9] Arlazarov, V.L., Arlazarov, V.V., Bulatov, K.B., Chernov, T.S., Nikolaev, D.P., Polevoy, D., Sheshkus, A.V., Skoryukina, N.S., Slavin, O.A., & Usilin, S.A. (2022). Mobile ID Document Recognition Coarse-to-Fine Approach. Pattern Recognition and Image Analysis 32, 89-108

[10] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database", 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255.

[11] He, Kaiming; Zhang, Xiangyu; Ren, Shaoqing; Sun, Jian (10 Dec 2015). Deep Residual Learning for Image Recognition. arXiv:1512.03385.

[12] Deng, Jia; Dong, Wei; Socher, Richard; Li, Li-Jia; Li, Kai; Fei-Fei, Li (2009). "ImageNet: A large-scale hierarchical image database". CVPR.

[13] Gemstones Images dataset, available at: https://www.kaggle.com/datasets/lsind18/gemstones-images/ (Accessed: October 01, 2023).

[14] R. F. Berriel, A. T. Lopes, A. F. de Souza, and T. Oliveira-Santos, “Deep Learning Based Large-Scale Automatic Satellite Crosswalk Classification,” IEEE Geoscience and Remote Sensing Letters, vol. 14, pp. 1513–1517, Sept 2017.

[15] R. F. Berriel, F. S. Rossi, A. F. de Souza, and T. Oliveira-Santos, “Automatic Large-Scale Data Acquisition via Crowdsourcing for Crosswalk Classification: A Deep Learning Approach,” Computers & Graphics, vol. 68, pp. 32–42, Nov 2017.

[16] "Rethinking the Inception Architecture for Computer Vision" - Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna (2016)

[17] J. Deng, W. Dong, R. Socher, L. -J. Li, Kai Li and Li Fei-Fei, "ImageNet: A large-scale hierarchical image database", 2009 IEEE Conference on Computer Vision and Pattern Recognition, 2009, pp. 248-255.

[18] "Information Theory, Inference, and Learning Algorithms" - David J.C. MacKay, 2003, pp. 604.

[19] "Deep Residual Learning for Image Recognition" - Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016)

[20] "Ensemble Methods in Data Mining: Improving Accuracy Through Combining Predictions" - Giovanni Seni and John Elder, 2010, pp. 352.

Исследование вопроса распознавания светофоров

В. Л. Лим
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2305378@edu.misis.ru

***Аннотация* — данная статья посвящена актуальной проблеме детектирования транспортных светофоров, которая играет всё более значимую роль в современной инфраструктуре и транспортной безопасности, поскольку все большее распространение получают идеи о создании беспилотных автомобилей. Управление беспилотным автомобилем предполагает решение задач, связанных с распознаванием объектов дорожной обстановки: дорог, пешеходов, автомобилей, препятствий, дорожных знаков, разметки, светофоров и т. д. Одной из ключевых задач является детектирования транспортных светофоров, что представляет собой процесс выделения различных элементов и компонентов данного дорожно-технического устройства, таких как корпус светофора и основные цветовые сигналы с целью анализа, и определения возможности движения в различных направлениях и навигации. В данной работе рассматривается два решения с открытым исходным кодом, основанных на архитектурах: YOLOv8, Faster R-CNN. И проведено сравнение их распознавания дорожных светофором на изображениях. В качестве входных данных используются изображения, полученные с камер установленных на автомобилях из следующих наборов данных: LISA Trafic Light Dataset, KAGGLE Traffic Light Detection Dataset.**

Ключевые слова — Компьютерное зрение, Детекция дорог, светофоров, Распознавание светофоров, Монокулярная камера, Нейронные сети, YOLOv8, Faster R-CNN, LISA Trafic Light Dataset, KAGGLE Traffic Light Detection Dataset.

I. Введение

В данной научной статье рассматривается актуальный вопрос проектирования систем беспилотных транспортных средства различного класса: от трамваев и поездов, до полноценных автомобилей. Данной проблеме уделяется немало внимания со стороны различных университетов, а также научно-исследовательских центров отдельных промышленных компаний, в том числе и в России (Яндекс, Сбер, Почта России, КамАЗ, Минтранс и др.) [1].

Для полноценного функционирования непилотируемой системы управления транспортом необходимо корректно обнаруживать и распознавать дорожно-транспортные объекты для правильной оценки ситуации на дорожном полотне. В качестве решения данной задачи наиболее предпочтительными инструментами являются технологии компьютерного зрения и нейронных сетей, особенно в контексте обработки видеопотока с камер, установленных на беспилотных автомобилях.

Специфический аспект рассматриваемой задачи заключается в обнаружении и распознавании сигналов светофоров, включая определение их текущего состояния (красный, зелёный или жёлтый). Литературный обзор охватывает разнообразные методы решения данной задачи [2], включая те, которые эффективны в условиях неблагоприятных погодных условий, таких как слабая освещённость, дымка, туман и другие [3].

В работе проводится анализ и сравнение современных методов глубокого обучения, известных своей высокой производительностью и способностью к обобщению в различных областях, таких как классификация [4] [5] и обнаружение объектов [6]. Особое внимание уделяется YOLO [7] и R-CNN [8] — двум современным детекторам, хорошо зарекомендовавшим себя в области задач, связанных с дорожным движением, такими как обнаружение светофоров и дорожных знаков.

Данная статья также оценивает текущие достижения в области глубокого обучения для определения местоположения и распознавания светофоров по изображениям 2D-камер с использованием открытых исходных кодов. Важным аспектом является не только доступность зарубежных баз изображений с аннотациями светофоров [9, 10, 11], но и необходимость высоких вычислительных мощностей для успешной реализации подходов, основанных на обучении.

II. Наборы данных

В ходе экспериментальных и обучающих процессов анализа рассматриваемых нейронных сетей привлекались разнообразные наборы данных, включая те, которые были созданы локально и подготовлены исследователями данного проекта, а также общедоступные выборки данных. В данной стадии исследования предлагается детальный анализ использованных общедоступных наборов данных.

A. LISA Traffic Light Dataset

Набор данных LISA [12] Traffic Light Dataset (LISA-TLD) (Рисунок 1. и Рисунок 2.) был создан в рамках исследовательской работы, проведенной Laboratory for Intelligent and Safe Automobiles (LISA) в Университете Калифорнии в Сан-Диего. Проект был инициирован в конце 2000-х годов. База данных собрана в городе Сан-Диего, штат Калифорния, США. База данных включает непрерывные тестовые и тренировочные видеопоследовательности, общим объемом 43 007 кадров и 113 888 аннотированных светофоров. Последовательности снимаются стереокамерой, установленной на крыше транспортного средства, движущегося как днем, так и ночью при различных условиях освещенности и погоде. В этой базе данных используется только левый вид камеры, и поэтому стереофункция в настоящее время не используется.

Рисунок 1. Пример тестового изображении базы данных LISA Traffic Light Dataset

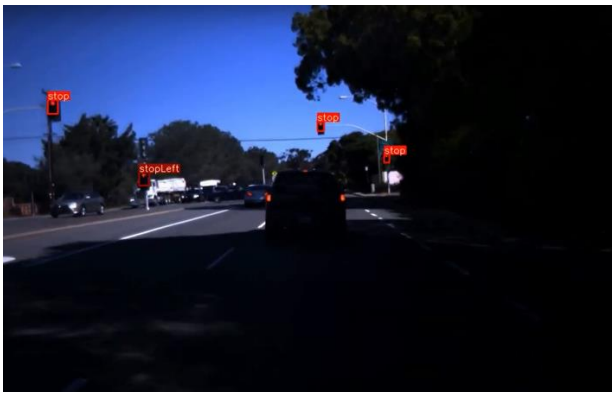


Рисунок 2. Пример результата на тестовом изображении базы данных LISA Traffic Light Dataset

B. KAGGLE Traffic Light Detection Dataset

Этот набор данных (Рисунок 3.) основан на CCF BDCI и используется для предсказания светофоров [13]. Набор данных содержит 2600 искусственно размеченных категорий светофоров и цветов. Категории светофоров включают следующие 9 категорий:

- Светофор транспортного средства
- Светофор для немоторизованных транспортных средств
- Светофор для левого поворота немоторизованных транспортных средств
- Светофор для пешеходных переходов
- Световые индикаторы полосы движения
- Световой индикатор направления
- Мигающий предупредительный свет
- Световой индикатор перехода
- Световой индикатор разворота

Визуализация данных и аннотаций, следующая: световые сигналы одного цвета визуализируются одним и тем же цветом.

Рисунок 3. Пример данных базы данных KAGGLE

III. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. Ultralitics You Only Look Once version 8 (YOLOv8)

В работе [14] решается задача распознавания дорожно-транспортных светофоров для применения в беспилотных автомобилях. Авторы предлагают использовать детектирующую нейронную сеть, локализующую и распознающую светофоры в кадре в реальном времени.

Архитектура YOLOv8 [15], также известная как "You Only Look Once" версии 8, представляет собой эволюцию в серии моделей, предназначенных для задач обнаружения объектов.

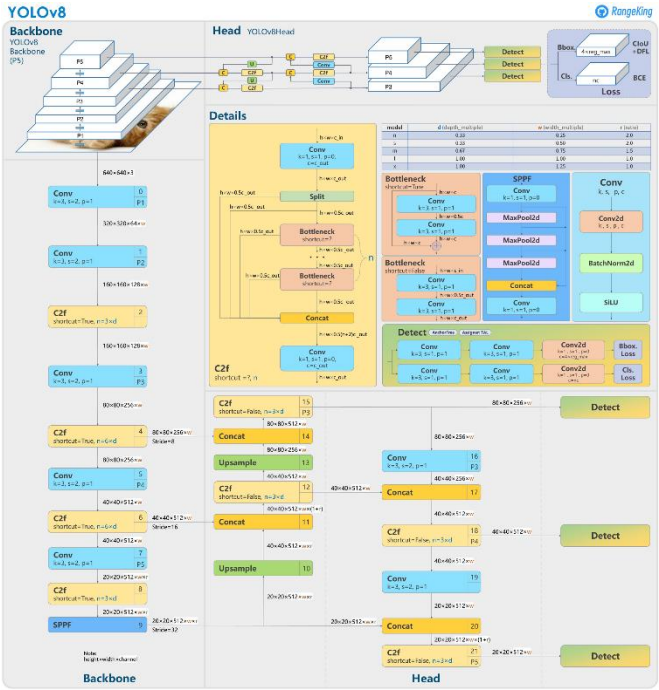


Рисунок 4. Архитектура YOLOv8, воссозданная пользователем *RangeKing*.

В основе архитектуры YOLOv8 (Рисунок 4.) [22] лежит концепция единого просмотра изображения, где обработка происходит в едином проходе через сеть. Несмотря на отсутствие формальных публикаций, некоторые особенности архитектуры YOLOv8 можно выделить на основе доступной информации:

- **Архитектурные изменения**: YOLOv8 внесла значительные изменения по сравнению с предыдущими версиями (YOLOv5-v4). Эти изменения включают в себя улучшения в структуре нейронной сети, оптимизации и новые слои для более эффективного обучения.
- **Механизм внимания**: В YOLOv8 внедрены механизм внимания. Механизм внимания представляет собой технику, которая позволяет сети сосредотачиваться на определенных частях входных данных, что может быть полезным для задач обнаружения объектов.

B. Faster R-CNN

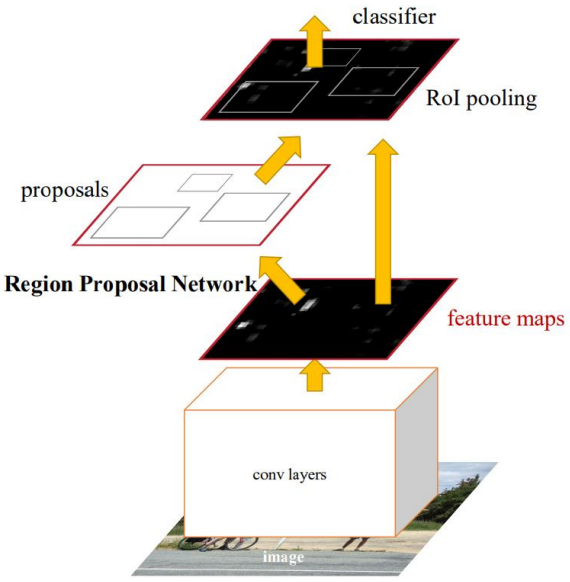


Рисунок 5. Архитектура Faster R-CNN с модулем RPN

Faster R-CNN — модель глубокого обучения для обнаружения объектов в изображениях, представленная в 2015 году. Она включает в себя ключевой элемент, такой как Region Proposal Network (RPN), работающий параллельно с основной сетью для предсказания областей изображения, где могут находиться объекты.

Основная сеть Faster R-CNN, часто использующая сверточные нейронные сети (CNN) [17] как VGG16 [18] или ResNet, обрабатывает предложенные области от RPN для более точного обнаружения объектов. Архитектура Faster R-CNN работает в два этапа: сначала генерируются предложения RPN [19], затем эти предложения используются для обнаружения объектов в каждой областиРисунок 4. (Рисунок 5.).

Одним из преимуществ Faster R-CNN является высокая точность обнаружения, обеспечиваемая использованием RPN. Модель обучается end-to-end, что упрощает процесс обучения и улучшает конечные результаты. Тем не менее, Faster R-CNN требует значительных вычислительных ресурсов, что может ограничивать его применение в ресурсоограниченных средах. Вопреки названию "Faster", эта модель может быть несколько медленнее по сравнению аналогами, такими как YOLO. В целом, Faster R-CNN представляет важный вклад в область обнаружения объектов, демонстрируя способность объединения областных предложений и обнаружения объектов в единой модели для улучшения производительности.

IV. Сравнение

Проведем визуальный анализ полученных результатов на Рисунок 6. и Рисунок 7. .

а) 

б) 

в) 

г) 

д)

Рисунок 6. Полученные результаты используя нейросеть YOLOv8.

а)

б)

в)

г)

д)

Рисунок 7. Полученные результаты используя нейросеть Faster R-CNN.

Для оценки сравнения работы двух нейронных сетей были проведены тесты с использованием наборы данных LISA Trafic Light Dataset – 5048 изображений с 14291 светофором.

На основе представленных результатов на Рисунок 6. и Рисунок 7. а можно визуально оценить качество работы двух нейронных сетей и констатировать, что обе нейронные сети показывают удовлетворительный результат. У каждой есть свои недостатки. Например, в то время как YOLOv8 определяет зеленый автобус как светофор на Рисунке 6 б) и распознает светофоры, расположенные вдалеке на Рисунке 6 а) (даже светофор, повернутый от водителя), то Faster R-CNN на аналогичных изображениях четко определяет светофоры, расположенные на контрастном фоне, но не распознает те, что расположены на фоне менее контрастных объектов, например зданий или иных элементов дорожно-транспортной обстановки.

Для лучшей проверки работоспособности и эффективности применения нейронных сетей, были использованы дополнительные метрики. Поэтому была применена система оценки метриками путем вычисления коэффициента Жаккара (Intersection over Union, IoU) для каждой обнаруженной детекции [20, 21]. В нем реализуется ассоциация обнаруженных светофоров с имеющейся разметкой при пороге в 10%. Вводятся следующие параметры:

- TP (True Positive) – детектор корректно локализовал светофор (прямоугольники разметки

и детекции пересекаются более, чем на 10%, относительно их общей площади).

- FP (False Positive) – детектор обнаружил светофор там, где его не было, то есть не найдено такого прямоугольника в разметке кадра, который пересекался бы с обнаруженным более, чем на 10%.
- FN (False Negative) – детектор не обнаружил светофор, хотя он присутствует, и у него есть разметка – пересечение менее, чем на 10%.

Следует отметить, что в данном контексте величина TN (True Negative) не определена, поскольку она предполагает, что детектор не обнаружил светофор там, где его действительно нет. На основе введенных параметров вычисляются следующие оценочные функции:

$Precision = \frac{TP}{TP+FP}$ – частота обнаружения светофора детектором в местах его реального наличия, относительно общего числа предсказанных светофоров;

$Recall = \frac{TP}{TP+FN}$ – количество обнаруженных детектором светофоров из действительно присутствующих в кадрах;

$F1 = 2\frac{Precision * Recal}{Precision + Recal} = \frac{2TP}{2TP+FP+FN}$ – оценка баланса между точностью (Precision) и полнотой (Recall).

ТАБЛИЦА 1. Показатели частоты верного обнаружения

Нейронная сеть	Precision
YOLOv8	0,85
Faster R-CNN	0,89

Для начала сравним параметр Precision. Как видно из Таблицы 1 модель YOLOv8 показала результат в 0.85. чуть лучший результат точности обнаружения светофоров. С другой стороны, Faster R-CNN, показавшая себя с лучшей стороны, показала немного более высокий результат Precision, равный 0.89.

ТАБЛИЦА 2. Показатели частоты обнаружения из общего количества светофоров

Нейронная сеть	Recall
YOLOv8	0,86
Faster R-CNN	0,91

Теперь сравним показатели полноты обнаружения светофоров Recall. Таблица 2 констатирует, что модель YOLOv8 показа результат Recall в 0.86, что также является чуть меньшим показателем относительно другой модели Faster R-CNN, которая, в свою очередь набрала 0.91 в оценке полноты детекции.

ТАБЛИЦА 3. Показатели баланса между точностью и полнотой обнаружения

Нейронная сеть	F1
YOLOv8	0,87
Faster R-CNN	0,90

Таблица 3 подводит итоги, закрепляя предыдущие показатели рассматриваемых нейронных сетей по показателям Precision и Recall, указывая на то, что модель YOLOv8 получила оценку баланса между точностью и полнотой распознавания величиной в 0.87, что на 0.03 пункта меньше, чем Faster R-CNN.

V. Заключение

Были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети. Приведены две нейронные сети для сегментации дорожного покрытия: YOLOv8 и Faster R-CNN. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведённые подходы протестированы на тестовом наборе из LISA Traffic Light Dataset. Отдельно были проведены визуальный анализ и используя метрику точности. По полученным результатам, очевидно, что нейронная сеть Faster R-CNN превосходит YOLOv8 по всем трем оцениваемым меркам (Precision, Recall, F1). При проведении сравнительного анализа уделялось внимание конкретным моделям с учетом конкретных параметров весов. Для комплексного сопоставления архитектур важны фиксированные комплекты данных и методологии обучения и тестирования.

Литература

[1] Перспективное развитие ИТС: беспилотный транспорт, 2022 https://www.tadviser.ru/index.php/Статья:Перспективное_развитие_ИТС:_беспилотный_транспорт

[2] Jensen M.B., Philipsen M.P., Møgelmose A., Moeslund T.B., Trivedi M.M. “Vision for looking at traffic lights: Issues, survey, and perspectives”, IEEE Transactions on Intelligent Transportation Systems, 2016, no. 17, pp. 1800–181.

[3] М. Г. Гродничев, М. С. Мосева «Исследование влияния тумана на системы машинного зрения» Московский технический университет связи и информатики, Москва 2022.

[4] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[5] Tanchenko, A.P., Fedulin, A.M., Bikmaev, R.R. et al. UAV Navigation System Autonomous Correction Algorithm Based on Road and River Network Recognition. Gyroscopy Navig. 11, 293–299 (2020). https://doi.org/10.1134/S2075108720040100

[6] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[7] Д. В. Никитин, И. С. Тараненко, А. В. Катаев, «Детектирование дорожных знаков на основе нейросетевой модели YOLO» Волгоградский государственный университет, Волгоград 2023.

[8] А. В. Игнатьев, М. А. Куликов, Д. Н. Цапиев, В. В. Тырин, «Методика автоматической классификайии дорог с использованием нейронной сети Mask R-CNN» Волгоградский государственный технический университет, Волгоград 2022.

[9] Jensen M.B., Philipsen M.P., Møgelmose A., Moeslund T.B., Trivedi M.M. “Vision for looking at traffic lights: Issues, survey, and perspectives”, IEEE Transactions on Intelligent Transportation Systems, 2016, no. 17, pp. 1800–1815.

[10] Behrendt K., Novak L., Botros R. “A deep learning approach to traffic lights: Detection, tracking, and classification”, 2017 IEEE International Conference on Robotics and Automation (ICRA), 2017, pp. 1370–1377.

[11] Yu F., Xian W., Chen Y., Liu F., Liao M., Madhavan V., Darrell T. “Bdd100k: A diverse driving video database with scalable annotation tooling”, arXiv preprint arXiv, 2018, 1805.04687.

[12] База данных LISA Trafic Light Dataset https://www.kaggle.com/datasets/mbornoe/lisa-traffic-light-dataset/data

[13] База данных KAGGLE Traffic Light Detection Dataset https://www.kaggle.com/datasets/wjybuqi/traffic-light-detection-dataset

[14] Possatti, Lucas C. et al. “Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars”, 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-8.

[15] Le Ba Chung, Duc Duy Nguyen, “Real-Time object detection and tracking for mobile robot using YOLOv8 and strong sort”, Vietnam Hanoi 2023.

[16] Ya. S. Pikalyov, T. V. Yermolenko “About neural architectures of feature extraction for the problem of object recognition for the problem of object recognition on devices with limited computing power”, Donetsk 3.30.2023.

[17] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” CVPR, 2015.

[18] A. I. Gavrilov, M. Tr. Do, “Classification of weld defects based on convolution neural network”, Bauman Moscow State Technical Univercity, Moscow, Russian Federation 2021.

[19] М. С. Тимошкин, А. Н. Миронов, А. С. Леонтьев, «Сравнение YOLOv5 и Faster R-CNN для обнаружения людей на изображении в потоковом режиме», МИРЭА Российский технологический университет, Москва, Россия 2022.

[20] Z. C. Lipton, C. P.Elkan, B. Narayanaswamy. “Thresholding Classifiers to Maximize F1 Score”, 2014 arXiv: Machine Learning, pp. 1-16.

[21] M. Sokolova, N. Japkowicz, S. Szpakowicz. “Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation”, Proceedings of Australasian joint conferenceon artificial intelligence, 2006, vol. 4304, pp. 1015-1021.

[22] D. Wu, M. Liao, W. Zhang, X. Wang, “YOLOP: You Only Look Once for Panoptic Driving Perception,” arXiv:2108.11250, 2022.

Вопросы сегментации дорожного слоя

А. А. Абакумов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2305400@edu.misis.ru

В. О. Хуако
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
v.khuako@edu.misis.ru

***Аннотация* — данная статья посвящена актуальной проблеме сегментации дорожного слоя, которая играет всё более значимую роль в современной инфраструктуре и транспортной безопасности. Всё большее распространение получают идеи о создании беспилотных автомобилей. Управление беспилотным автомобилем предполагает решение задач, связанных с распознаванием объектов дорожной обстановки: дорог, пешеходов, автомобилей, препятствий, дорожных знаков, разметки, светофоров и т. д. Одной из ключевых задач является сегментация дорожного слоя, что представляет собой процесс выделения различных элементов и компонентов дорожного покрытия, таких как асфальт, бордюры, линии разметки и другие с целью анализа, и определения доступной зоны для выполнения маневров и навигации. В данной работе рассматривается два решения с открытым исходным кодом, основанных на архитектурах: HybridNets, YOLOP. И проведено сравнение их способностей в сегментация дорожного слоя на изображениях. В качестве входных данных используются изображения, полученные с камер установленных на автомобилях из следующих наборов данных: KITTY, BDD100K.**

Ключевые слова — Компьютерное зрение, Детекция дорог, Монокулярная камера, Беспилотные автомобили, YOLOP, mAP, BDD100K, KITTY, HybridNets.

I. Введение

В настоящее время, в свете стремительного развития технологий, особое внимание уделяется исследованиям, посвященным разработке беспилотных систем. Данные исследования охватывают широкий спектр вопросов, таких как: разработка навигационных систем [1], разработка алгоритмов для автономной коррекции навигационных систем на основе распознавания дорожной и речной сети [2], визуальная локализация наземных транспортных средств с помощью монокамер и дорожных знаков с геодезическими границами[3], использование нейронных сетей для детектирования светофоров на изображениях [4], повышение точности сопровождения подвижных объектов с помощью алгоритма комплексной обработки сигналов с монокулярной камеры и LIDAR [5] и др.

В данной статье будет рассмотрен один из аспектов разработки беспилотных транспортных средств, а именно автомобилей, проектированием которых занимаются многие промышленные компании в области автомобилестроения (Tesla, КАМАЗ, Ford, Volkswagen, Mercedes и т. д.) и ИТ-компании (Яндекс, Google и т. д.) при поддержке университетов и научно-исследовательских центров [6].

Сегментация дороги, являясь одним из основных модулей, что воспринимает окружающую обстановку и определяет область, пригодную для движения. Пригодным для движения регионом является связный участок поверхности дороги, не занятый транспортными средствами, пешеходами, велосипедистами и другими препятствиями.

Сегментация дорог с помощью камер исследуется уже несколько десятилетий, поскольку камеры часто генерируют кадры высокого разрешения и являются экономически эффективными. Традиционные алгоритмы компьютерного зрения использовали для сегментации дорог признаки, определяемые вручную, такие как края и гистограммы. Однако эти признаки работали в ограниченных ситуациях, и их было трудно распространить на новые сценарии.

В последние годы интерес исследователей привлекают алгоритмы на основе свёрточных нейронных сетей (CNN) [7]. Благодаря применению массивных сверточных CNN способны работать с различными сценариями движения. Существующие алгоритмы сегментации дорог на основе CNN, такие как FCN [8], SegNet [9], StixelNet [10] и Up-conv-Poly [11], позволяют получить точную область, пригодную для движения, но требуют больших вычислительных затрат.

II. Наборы данных

В рамках проведения экспериментов и обучения анализируемых нейронных сетей включались разнообразные наборы данных, в том числе как те, которые были составлены локально и подготовлены авторами данного исследования, так и общедоступные выборки данных. Подробно проанализируем использованные общедоступные наборы данных.

A. KITTY

Набор данных KITTY [12] был создан в сотрудничестве с Янником Фричем и Тобиасом Кюнлом из Honda Research Institute Europe GmbH. Набор данных состоит из 289 учебных и 290 тестовых изображений. Он содержит три различные категории дорожных сцен (Рис.1):

- uu — городские немаркированные (98/100)
- um — городская разметка (95/96)
- umm — городские многополосные дороги с разметкой (96/94)
- urban — комбинация трех вышеперечисленных категорий.

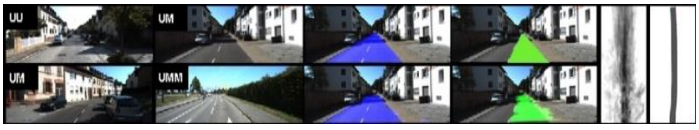


Рис. 1. Категории дорожных сцен и сегментация дороги.

Истинные данные были получены путем ручного анализа изображений и доступны для двух различных типов дорожного рельефа:

- road — область дороги, т. е. состав всех полос движения.

- lane — это-полоса, т. е. та полоса, по которой в данный момент движется автомобиль (доступно только для категории "um").

Истина предоставляется только для учебных изображений.

Следующие изображения, представляют качественную иллюстрацию работы метода на нескольких образцах тестовых изображений. Первоначально, приводятся результаты анализа перспективного изображения (Рис.2), после чего проводится оценка метода на изображении, полученном с высоты птичьего полета (Рис.3).

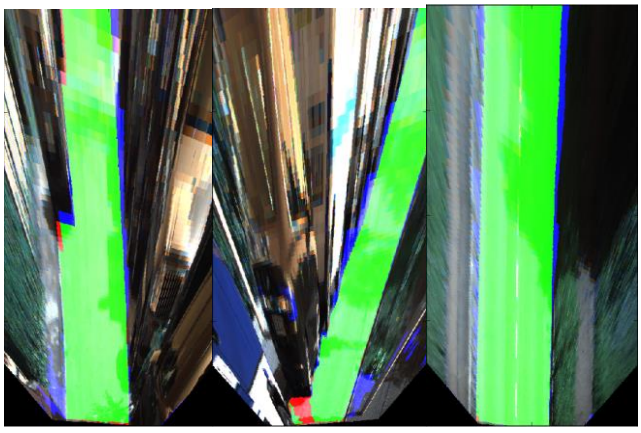
- Красным цветом обозначены ложноотрицательные результаты.
- Синие области соответствуют ложноположительным результатам.
- Зеленые - истинно положительным.

а

б

в

Рис. 2. Примеры результатов на перспективном изображении.



а б в

Рис. 3. Примеры результатов с высоты птичьего полета

B. BDD100K A Large-scale Diverse Driving Video Database

Набор данных BDD100K [13] содержит. набор данных из 100 000 видеороликов. Каждое видео длится около 40 секунд, имеет разрешение 720p и частоту 30 кадров в секунду, что в общем предоставляет 120 миллионов изображений. Видеоролики были собраны в различных районах США и предоставляет изображения отражающие разнообразные аспекты дорожного движения, местности как в городских, так и в сельских районах, в разные времена суток и при различных погодных условиях.

В наборе данных выбирается ключевой кадр на 10-й секунде из каждого видео и прикрепляются аннотации к этим ключевым кадрам. Они маркируются на нескольких уровнях: метки изображения, ограничительные рамки дорожных объектов, зоны движения, разметка полосы движения, сегментация экземпляров на весь кадр. Эти аннотации помогут понять разнообразие данных и статистику объектов в различных типах сцен, примеры подобных кадров представлены на Рис. 4 и Рис. 5.

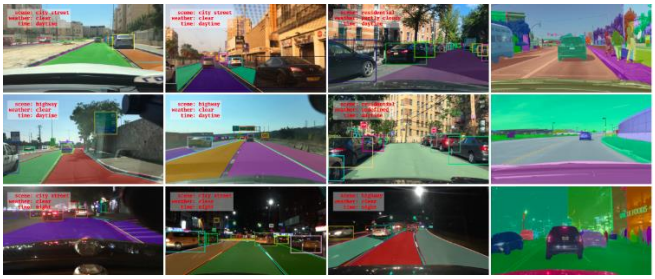
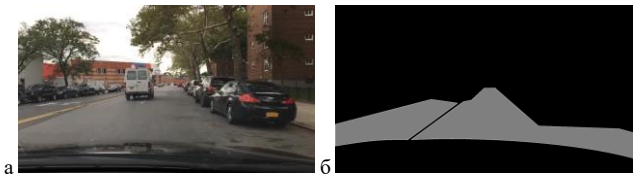


Рис. 4. Пример изображений из набора BDD100K с наложенными масками сегментации дорог, дорожными полосами, обнаруженными объектами.



а б

в г

Рис. 5. Пример изображений из набора BDD100K с наложенными масками сегментации дорог.

III. Нейросетевые архитектуры

A. HybridNets End2End Perception Network

В работе [14] решается задача распознавания дорожных объектов, сегментации дорожного покрытия и обнаружения полос разметки дорог в режиме реального времени для применения во встроенных системах беспилотных автомобилей. Авторы предлагают использовать сквозную сетевую архитектуру HybridNets (Рис. 5).

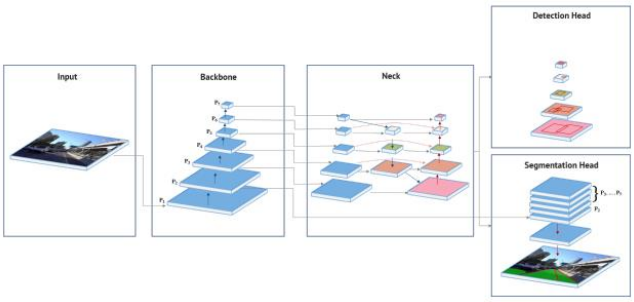


Рис. 5. Архитектура HybridNets

В основе архитектуры HybridNets лежит сверточная нейронная сеть (CNN) EfficientNet-B3[15] с функцией масштабирования для увеличения глубины сети, количества фильтров и разрешения изображений для оптимизации вычислений, извлекающая признаки из изображений путем поиска локальных зависимостей в текстурах, цветах, гранях и других объектов. HybridNets использует зависимость, где каждый P_{i+1} уровень слоя имеет разрешения в два раза меньше, чем P_i. Всего уровней P_7, где P_1 входное изображение. Полученные признаки поступают на “горлышко”(Neck) нейронной сети, представленное модулем BiFPN [16] основанным на EfficientDet, что объединяет признаки с разным разрешением на основе межмасштабной связи для каждого узла по каждому двунаправленному пути и добавляет вес для каждого признака. EfficientNet-B3 и BiFPN составляют один общий кодер.

Декодер же делится на две независимые части. Первая - сеть обнаружения заимствует кластеризацию k-means [17] из YOLOv4 [18]. Для каждой ячейки сетки выбираются 9 кластеров с тремя различными масштабами и предсказывается смещения ограничительных полей, вероятность каждого класса, а также уверенность предсказанный полей. Вторая - сеть сегментации имеет три класса вывода: зона дорожного покрытия, полосы разметки дороги и остальной фон. В процессе сегментации на вход поступают уровни признаков из “горлышка” и масштабируются до размера уровня P_2, после чего суммируются и представляют карту признаков показывающую принадлежность каждого пикселя к классу. Затем дополнительно сравнивается с картой признаков P_2 из основной сети, что улучшает точность.

HybridNets обучалась на наборе данных BDD100K используя тренировочный набор из 70 тыс. изображений, валидационный - из 10 тыс. и тестовый - из 20 тыс. изображений. Поскольку метки тестового набора не были доступны, оценка сети проводилась на валидационном наборе. Размер изображений соответствовал значениям 640×384, как компромисс между скоростью и качеством.

Обучение HybridNets происходило по алгоритму:

- Обучается кодер и сеть обнаружения.
- Замораживаются параметры кодера и сеть обнаружения, обучается сеть сегментации.
- Сеть обучается совместно для всех задач.

B. YOLOP

В исследовании [19] авторы представляют схожий подход для решения аналогичных задач обнаружения и сегментации, используя сеть паноптического восприятия движения YOLOP ("You Only Look Once for Panoptic").

Архитектура YOLOP (Рис. 6) в отличие от HybridNets включает в себя один кодер для извлечения признаков и три декодера каждый из которых предназначен для решения конкретных задач.

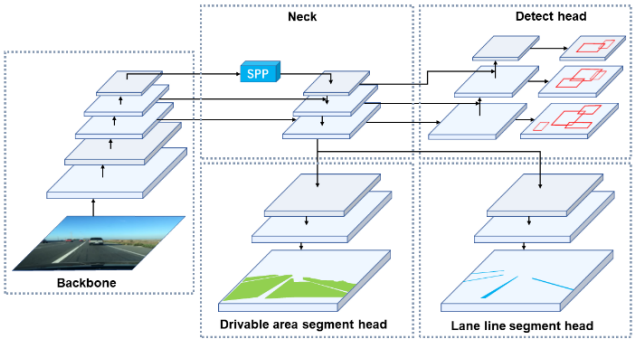


Рис. 6. Архитектура YOLOP

Авторы YOLOP вдохновляясь выдающийся производительностью YOLOv4, выбрали использовать в основе для извлечения признаков входного изображения CSPDarknet, что решает проблему дублирования градиентов и поддерживает передачу и повторное использование признаков, что снижает количество параметров и вычислений, повышая быстродействие.

“горлышко” (Neck) состоит из модуля пространственного пирамидального объединения (SPP)[20], что генерирует и объединяет признаки разных масштабов, и модуля пирамидальной сети признаков (FPN)[21], что объединяет признаки разных семантических уровней, благодаря чему генерируемые содержат информацию разных масштабов и разных семантический уровней. YOLOP использует метод конкатенации для объединения признаков.

Сеть обнаружения в YOLOP так же, как и HybridNets заимствует кластеризацию k-means из YOLOv4 и решает аналогичную задачу распознавания дорожных объектов.

Сеть сегментации дорожного покрытия и сеть сегментации полос движения используют одну структуру сети, где в сеть подаётся три карты признаков из модуля FPN и масштабируются до наибольшего среди этих уровней масштаба, что предсказывает вероятность принадлежности каждого пикселя к классу. Благодаря общему SPP модулю в “горлышке” отсутствует дополнительный SPP модуль для сетей сегментации, обосновывая, что это не принесет увеличения производительности. Однако для увеличения производительности используется метод ближайшей интерполяции в слое масштабирования вместо обратной свертки.

YOLOP обучалась на наборе данных BDD100K. Размер изображений соответствовал значениям 640×384.

YOLOP использует следующий алгоритм обучения:

- обучается кодировщик и сеть обнаружения
- замораживаются кодер и сеть обнаружения, и обучаются две сети сегментации
- обучается вся сеть совместно для всех трех задач

IV. Сравнение

Проведем визуальный анализ полученных результатов на Рис. 7 и 8.

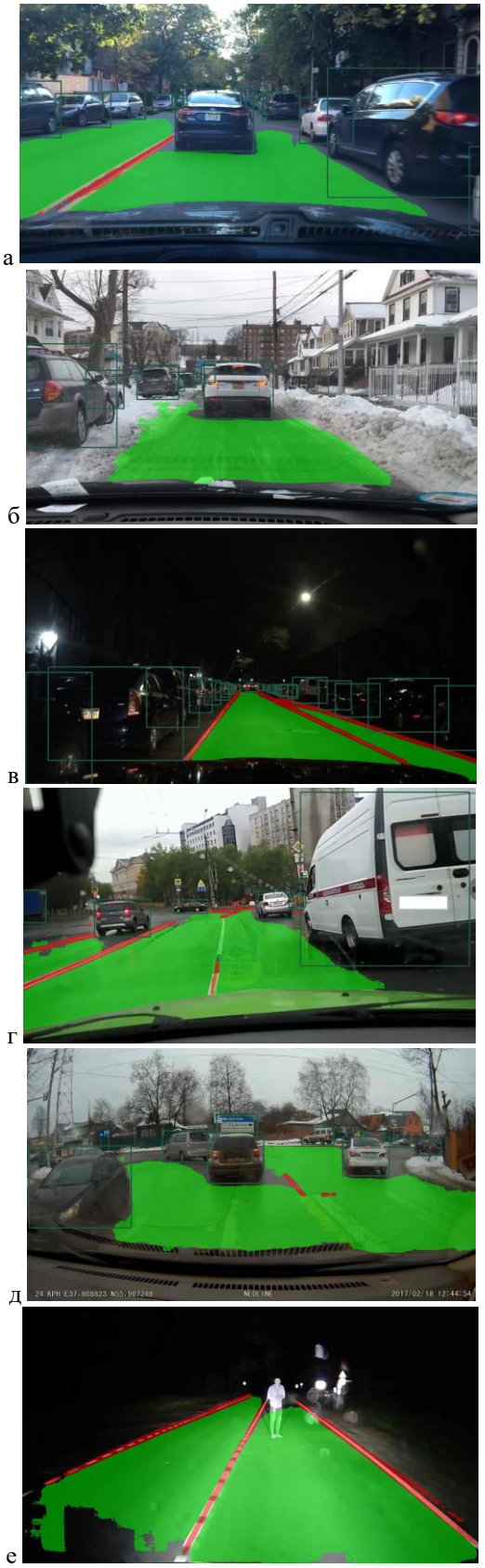


Рис. 7. Полученные результаты используя нейросеть HybridNets.

Рис. 8. Полученные результаты используя нейросеть YOLOP.

Исходя из представленных результатов можно выделить, что обе нейронные сети удовлетворительно распознают дорожные объекты, дорожное покрытие, и дорожную разметку, но при наличии необычных ситуаций таких как человек на дороге (Рис. 7,8 (e)) и перекрестки начинают происходить ошибки. Также стоит отметить, что при наличии человека на проезжей части YOLOP смогло распознать человека, как препятствие, то HybridNets не смогло этого достичь и обозначило, как доступное для передвижения пространство. В свою очередь HybridNets чуть лучше справляется с распознаванием дорожного покрытия на полосе встречного движения и перекрестках.

Проведем так же сравнение технических параметров, в контексте их производительности в задаче обнаружения объектов на изображениях. Для оценки эффективности каждой модели были использованы 1000 изображений из набора данных BDD100K.

ТАБЛИЦА 1. Оценка точности

Модель	Точность (Accuracy)
HybridNets	0.938
YOLOP	0.968

Первым критерием для сравнения является точность обнаружения объектов. YOLOP продемонстрировала впечатляющую точность с показателем Accuracy (Acc) равным 0.968. С другой стороны, HybridNets, хотя и обладает высокой точностью, показала немного более низкий результат с Acc равным 0.938. Визуальный анализ подкрепляет метрику оценки точности в том, что нейронные сети примерно на одном уровне в точности определения дорожного покрытия.

ТАБЛИЦА 2. Оценка по индексу пересечения (IoU)

Модель	IoU
HybridNets	0.826
YOLOP	0.830

Индекс пересечения (IoU) параметр, определяющий степень соответствия обнаруженных объектов на изображении и действительных объектов. В данном сравнении YOLOP имеет высокие показатели с IoU равным 0.830, в то время как HybridNets демонстрирует результаты немного ниже с IoU равным 0.826.

ТАБЛИЦА 3. Оценка среднего индекса пересечения (mIOU)

Модель	mIOU
HybridNets	0.966
YOLOP	0.896

Средний индекс пересечения (mIOU) представляет собой среднее значение IOU по всем объектам на изображении. HybridNets, демонстрирует mIOU на уровне 0.966, тогда как YOLOP, хоть и имеет показатель ниже, достигает значения mIOU равного 0.896.

V. Заключение

Были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети. Приведены две нейронные сети для сегментации дорожного покрытия: YOLOP и HybridNets. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведённые подходы протестированы на тестовом наборе из BDD100K. Отдельно были проведены визуальный анализ и используя метрики точности, индекса пересечения и среднего индекса пересечения. По полученным результатам, очевидно, что нейронная сеть HybridNets превосходит YOLOP в оценке среднего индекса пересечения (mIOU), и примерно одинаковы в оценке точности и оценке по индексу пересечения (IoU). При проведении сравнительного анализа уделялось внимание конкретным моделям с учетом конкретных параметров весов. Для комплексного сопоставления архитектур важны фиксированные комплекты данных и методологии обучения и тестирования.

Литература

[1] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[2] Tanchenko, A.P., Fedulin, A.M., Bikmaev, R.R. et al. UAV Navigation System Autonomous Correction Algorithm Based on Road and River Network Recognition. Gyroscopy Navig. 11, 293–299 (2020). https://doi.org/10.1134/S2075108720040100

[3] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[4] Толстенко Л.С., Клейменов А.А., Али Б., Крынецкая Г.С., Коробков А.А. Анализ нейронных сетей для детектирования светофоров на изображениях // известия института инженерной физики. — 2023. — № 2(68). — С. 59-65.

[5] R. R. Bikmaev, M. D. Zolotov, A. N. Popov and R. N. Sadekov, "Improving the Accuracy of Supporting Mobile Objects with the Use of the Algorithm of Complex Processing of Signals with a Monocular Camera and LiDAR," 2019 26th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2019, pp. 1-4, doi: 10.23919/ICINS.2019.8769360.

[6] “Driverless cars (global market)”, available at: https://www.tadviser.ru/index.php/Статья:Беспилотные_автомобили_(мировой_рынок) (Accessed: November 25, 2023).

[7] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard and L. D. Jackel: Backpropagation Applied to Handwritten Zip Code Recognition, Neural Computation, Winter 1989.

[8] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” CVPR, 2015.

[9] V. Badrinarayanan, A. Handa, and R. Cipolla, “Segnet: A deep convolutional encoder-decoder architecture for robust semantic pixel-wise labelling,” CoRR, 2015.

[10] Levi, D.; Garnett, N.; Fetaya, E.; Herzlyia, I. StixelNet: A Deep Convolutional Network for Obstacle Detection and Road Segmentation. In Proceedings of the 26th British Machine Vision Conference (BMVC), Swansea, UK, 1 January 2015.

[11] Li, F. et al. Fully convolutional pyramidal networks for semantic segmentation. IEEE Access 2020.

[12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, “Vision meets robotics: The kitti dataset,” The International Journal of Robotics Research, 2013.

[13] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, Trevor Darrell. "BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling"

[14] VT Data NVH Baoa PD Hunga FPT University, Hoa Lac High Tech Park, Hanoi, 10000, Vietnam

[15] M. Tan, Q.V. Le, “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks,” arXiv:1905.11946, 2020.

[16] M. Tan, R. Pang and Q. V. Le, "EfficientDet: Scalable and Efficient Object Detection," in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10778-10787, doi: 10.1109/CVPR42600.2020.01079.

[17] J.B. MacQueen, “Some Methods for Classification and Analysis of Multivariate Observations,” in Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics, University of California Press, Berkeley, 281-297, 1967.

[18] C. Y. Wang, A. Bochkovskiy, H. Y. M. Liao. ScaledYOLOv4: Scaling cross stage partial network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, IEEE, Nashville, USA, pp.13024–13033, 2021. DOI: 10.1109/CVPR46437.2021. 01283.

[19] D. Wu, M. Liao, W. Zhang, X. Wang, “YOLOP: You Only Look Once for Panoptic Driving Perception,” arXiv:2108.11250, 2022.

[20] K. M. He, X. Y. Zhang, S. Q. Ren, J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.37, no.9, pp.1904–1916, 2015. DOI: 10.1109/TPAMI.2015.2389824.

[21] T. Y. Lin, P. Dollár, R. Girshick, K. M. He, B. Hariharan, S. Belongie. Feature pyramid networks for object detection. In Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, IEEE, Honolulu, USA, pp.936–944, 2017. DOI: 10.1109/CVPR.2017.106.

Классификация транспортных средств компьютерным зрением

Леонов Иван Юрьевич
Университет МИСИС
m2311081@edu.misis.ru

Аннотация—В данной статье рассматриваются теоретические аспекты сверточных нейронных сетей и популярных предобученных нейронных сетей. Исследуется эффективность двух предобученных сверточных нейронных сетей, ResNet50 и VGG16, в задаче классификации транспортных средств. Обучение моделей проводится на стартовом датасете, включающем изображения автомобилей, автобусов, грузовиков и мотоциклов. Результаты оцениваются на новом датасете, позволяя проанализировать обобщающую способность моделей. По итогу выбирается лучшая модель.

Ключевые слова——классификация транспортных средств, компьютерное зрение, сверточные нейронные сети, предобученные сверточные нейронные сети, оценка модели.

I Введение

Исследователи и разработчики в области машинного обучения и компьютерного зрения активно занимаются созданием и внедрением современных технологий в различные сферы жизни общества. Одной из таких сфер является транспортная индустрия. В свете стремления к созданию автономных транспортных систем разрабатываются новые подходы, позволяющие осуществлять навигацию и принимать решения без участия человека [1, 2, 3, 4, 5]. Системы обнаружения и классификации транспортных средств помогают предотвращать аварии, выявляя различные типы транспортных средств, пешеходов и другие объекты.

В контексте городской мобильности модели классификации транспортных средств могут быть применены для оптимизации движения и управления потоками транспорта в городской среде. Эти решения способствуют эффективному использованию инфраструктуры, способствуя созданию интеллектуальных городских систем. Это особенно важно в условиях роста городского населения и увеличения числа транспортных средств.

В рамках статьи будет проведен теоретический обзор двух ведущих сверточных нейронных сетей, а именно ResNet50 и VGG16, и исследуем их применение в задаче классификации транспортных средств. В рамках статьи будут рассмотрены архитектурные особенности данных моделей и их влияние на процесс обучения и производительность. При этом особое внимание уделим предобученным весам, которые стали ключевым элементом в повышении качества обучения и обобщения.

В данной статье будут рассмотрены теоретические аспекты сверточных нейронных сетей, предобученных и с нулевой инициализацией весов.

В последующих разделах будет проведено сравнение двух популярных сверточных нейронных сетей, а именно ResNet50 и VGG16. Будет проведен анализ качества моделей с использованием функции потерь, точности и матрицы ошибок. Итогом исследования будет выбор лучшей модели для классификации транспортных средств.

II Сверточные нейронные сети

Сверточные нейронные сети (CNN) широко применяются для решения задач классификации изображений. Они показывают выдающуюся производительность благодаря своей способности автоматически извлекать иерархические признаки из входных изображений.

Вот основные шаги, которые сверточные нейронные сети выполняют при решении задачи классификации изображений [6].

1) Определение модели.

- Входной слой (Input Layer): изображение подается на входной слой сети. Каждый пиксель представляется значением интенсивности цвета (или оттенка серого) в случае монохромных изображений.
- Сверточные слои (Convolutional Layers): эти слои используют фильтры (ядра) для сканирования входного изображения и извлечения различных признаков. Фильтры обычно представляют собой небольшие матрицы весов, которые перемещаются по изображению, умножаются на значения пикселей и создают карты признаков.
- Функции активации (Activation Functions): после сверточных операций, к полученным значениям применяются нелинейные функции активации, такие как ReLU (Rectified Linear Unit). Это помогает внести нелинейность в модель и улучшить ее способность извлекать сложные паттерны.
- Пулинговые слои (Pooling Layers): пулинг уменьшает размер карт признаков, сохраняя при этом наиболее важные информационные аспекты. Обычно используются максимальный (Max Pooling) и средний пулинг (Average Pooling).
- Полносвязные слои (Fully Connected Layers): после прохождения скрытых слоев признаки агрегируются и подаются на полносвязные слои. Эти слои работают аналогично обычным нейронным сетям и выполняют окончательное преобразование признаков в выходы, связанные с конкретными классами.
- Функция потерь (Loss Function): модель выдает вероятности принадлежности к каждому классу. Функция потерь оценивает, насколько эти вероятности соответствуют фактическим меткам классов в обучающем наборе.

2) Обучение: Сеть обучается на обучающем наборе с использованием алгоритма обратного распространения ошибки (Backpropagation) и методов оптимизации, таких как стохастический градиентный спуск (SGD). Модель корректирует свои веса и параметры для минимизации функции потерь.

3) Тестирование и предсказание: После завершения обучения модель может быть использована для классификации новых изображений.

III Предобученные сверточные нейронные сети

Предобученные сверточные нейронные сети представляют собой модели глубокого обучения, которые были обучены на больших наборах данных для выполнения задачи классификации изображений. Эти модели обладают способностью выделения важных признаков из входных изображений и автоматически извлекать характеристики, такие как края, углы, текстуры и более сложные паттерны.

Процесс предобучения в случае сверточных нейронных сетей часто включает в себя обучение модели на большом наборе данных, таком как ImageNet, который содержит миллионы изображений, представляющих различные категории объектов. Обученные на таких данных, CNN способны обнаруживать иерархические и абстрактные признаки, что делает их мощными инструментами для решения разнообразных задач в области компьютерного зрения.

После этапа предобучения эти модели могут быть использованы в различных задачах без необходимости обучения с нуля. Такой подход, называемый трансферным обучением, позволяет использовать знания, полученные моделью во время обучения на одной задаче, для улучшения производительности на другой задаче.

В настоящее время популярны следующие предобученные сверточные нейронные сети:

- AlexNet;
- DenseNet;
- GoogLeNet;
- MobileNet;
- ResNet;
- VGG.

A. AlexNet

AlexNet – это сверточная нейронная сеть, разработанная в 2012 году именно для задачи классификации изображений. Эта сеть стала важным этапом в истории глубокого обучения, поскольку впервые показала, что глубокие сверточные нейронные сети могут превзойти традиционные методы в компьютерном зрении.

Нейронная сеть AlexNet была представлена командой исследователей из лаборатории Vision Research Group в Университете Торонто. Основные авторы – Алекс Крисссе, Илья Сосинский, Корбан Веддж и другие.

Вот некоторые ключевые особенности AlexNet [7, 8].

1) Архитектура: AlexNet состоит из пяти сверточных слоев, которые чередуются с пулинговыми слоями, за которыми следуют три полносвязных слоя. В то время это было революционным, так как предыдущие нейронные сети были намного менее глубокими.

2) Функции активации: в качестве функции активации в AlexNet используется функция ReLU (Rectified Linear Unit), что помогло уменьшить проблему исчезающего градиента и ускорило обучение.

3) Специализированные вычислительные блоки: AlexNet использует графический процессор (GPU) для ускорения обучения. Это также является одним из важных факторов, который внес свой вклад в успешность сети, так как использование GPU позволяет эффективнее обрабатывать большие объемы данных.

4) Dropout: в AlexNet был впервые применен метод Dropout, который помогает предотвращать переобучение. Dropout случайным образом "выключает" некоторые нейроны в процессе обучения, что способствует улучшению обобщения.

5) Local Response Normalization (LRN): этот слой выполняет нормализацию активаций внутри локальных областей, что может способствовать обучению более устойчивых и обобщенных признаков.

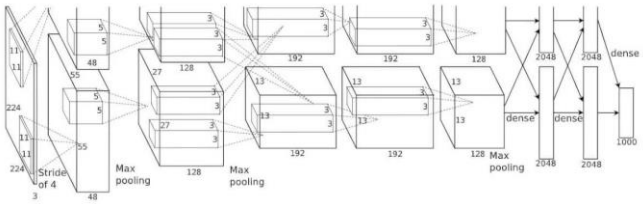


Рис. 1 – Архитектура AlexNet

AlexNet сыграла важную роль в развитии области глубокого обучения и сверточных нейронных сетей. Её архитектурные решения были в дальнейшем усовершенствованы в последующих моделях, таких как VGG, GoogLeNet и ResNet.

B. DenseNet

DenseNet, или Densely Connected Convolutional Networks, представляет собой архитектуру глубоких нейронных сетей, предназначенную для решения задач компьютерного зрения, таких как классификация изображений и сегментация объектов. Эта архитектура была предложена в статье "Densely Connected Convolutional Networks" в 2017 году Жифеном Хуангом, Янгом Лю и Ченг-Тао Ли.

Основная идея DenseNet заключается в том, чтобы каждый слой сети был связан с каждым предыдущим слоем, создавая плотные (dense) связи между слоями. Это отличается от традиционных сверточных нейронных сетей, где каждый слой подключен только к предыдущему и следующему слоям. В DenseNet каждый слой получает на вход все предыдущие слои, что позволяет эффективнее использовать признаки и облегчает передачу градиентов во время обучения.

Основные компоненты DenseNet [9, 10].

1) Dense Blocks: основной строительный блок DenseNet. Внутри блока каждый слой получает на вход все предыдущие слои, а выход каждого слоя передается всем следующим. Это создает плотные связи и способствует обмену информацией между слоями.

2) Transition Layers: используются для уменьшения размерности данных между блоками. Они включают сверточные слои и слои пулинга, чтобы уменьшить пространственные размеры признаков.

3) Global Average Pooling (GAP): в конце сети применяется слой глобального усреднения, который усредняет значения признаков по всем пространственным распределениям. Это позволяет сети иметь фиксированный размер выхода независимо от размера входного изображения.

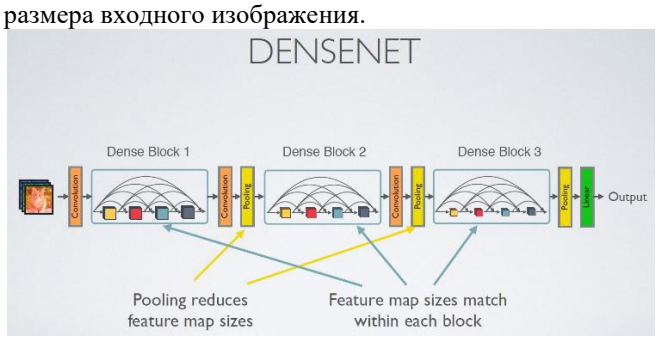


Рис. 2 – Архитектура DenseNet

DenseNet часто используется в задачах компьютерного зрения и показывает хорошие результаты на наборах данных для классификации изображений и сегментации объектов.

C. *GoogLeNet*

GoogLeNet, также известная как Inception, это глубокая нейронная сеть, разработанная исследовательским отделом Google, известным как Google Brain. Она была представлена в 2014 году на конференции по компьютерному зрению ImageNet.

Основной целью GoogLeNet было создание эффективной архитектуры глубокой нейронной сети, способной работать с большими изображениями и обладающей высокой точностью классификации. Особенностью этой сети является использование модулей, называемых "Inception modules", которые позволяют эффективно обрабатывать информацию на разных уровнях абстракции.

Основные характеристики GoogLeNet [11, 12].

1) Inception Modules: одним из ключевых элементов архитектуры являются Inception modules, которые представляют собой несколько параллельных сверточных слоев с различными размерами ядер (1x1, 3x3, 5x5) и слои с пулингом. Эти подмодули объединяются в один выходной тензор.

2) Глубина и эффективность: несмотря на свою глубокую архитектуру, GoogLeNet обладает меньшим количеством параметров по сравнению с некоторыми другими глубокими нейронными сетями, что делает ее более эффективной в плане использования ресурсов.

3) Глобальный средний пулинг (Global Average Pooling): в конце сети используется глобальный средний пулинг, который усредняет значения каждого признака по всей пространственной размерности. Это позволяет уменьшить количество параметров и предотвращает переобучение.

4) Auxiliary Classifiers: GoogLeNet также включает вспомогательные классификаторы перед несколькими скрытыми слоями. Эти вспомогательные классификаторы помогают ускорить обучение и предотвращают исчезновение градиентов.

5) Использование ReLU: в сети используется функция активации ReLU (Rectified Linear Unit) для ускорения обучения и предотвращения проблемы затухания градиента.

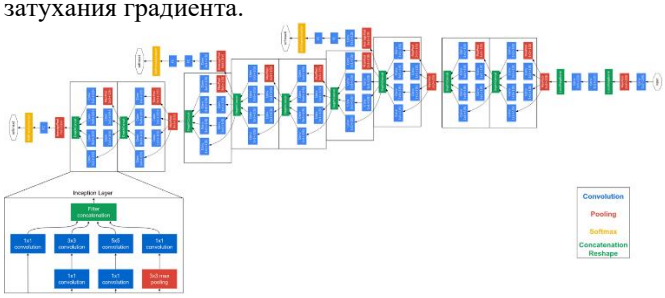


Рис. 3 – Архитектура GoogLeNet

GoogLeNet была успешной в соревнованиях по классификации изображений ImageNet, обеспечив высокую точность при более низком количестве параметров, чем предшествующие модели. Ее влияние привело к разработке других архитектур, таких как Inception V2, Inception V3 и других модификаций, улучшающих ее производительность.

D. *MobileNet*

MobileNet – это семейство легковесных нейронных сетей, специально разработанных для работы на мобильных устройствах с ограниченными ресурсами. Они были представлены в статье "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications", написанной исследователями Google в 2017 году.

Основная идея MobileNet заключается в том, чтобы создать эффективные сверточные нейронные сети с низким количеством параметров, которые могут работать на устройствах с ограниченной вычислительной мощностью и памятью, таких как смартфоны или встроенные системы.

Вот несколько ключевых аспектов MobileNet [13, 14].

1) Depthwise Separable Convolution: основной строительный блок MobileNet - это глубокая свертка (depthwise separable convolution), которая разделяет процесс свертки на два этапа: сначала применяется свертка по каждому каналу (depthwise convolution), затем применяется свертка 1x1 для объединения каналов (pointwise convolution). Это позволяет значительно снизить количество параметров и вычислений.

2) Width Multiplier: мобильные устройства обычно имеют ограниченные вычислительные ресурсы. Width Multiplier в MobileNet позволяет настраивать ширину (количество каналов) модели для баланса между точностью и вычислительной эффективностью. Значение Width Multiplier представляет собой коэффициент, который уменьшает количество каналов во всех слоях модели.

3) Resolution Multiplier: для того чтобы адаптировать модель под различные разрешения входных изображений, MobileNet предоставляет параметр Resolution Multiplier. Он регулирует размер входных изображений, что также влияет на количество вычислений и объем занимаемой памяти.

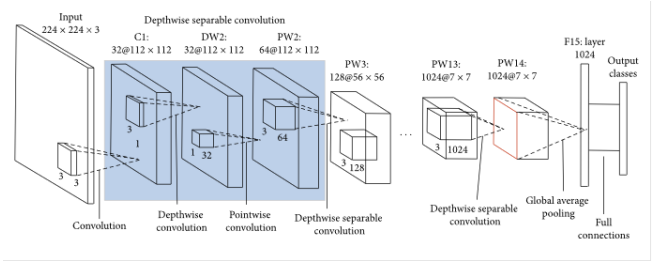


Рис. 4 – Архитектура MobileNet

MobileNet часто используется для задач компьютерного зрения на мобильных устройствах, таких как классификация изображений, детекция объектов и сегментация изображений.

MobileNet показывает хорошие результаты в условиях ограниченных ресурсов, делая их привлекательным выбором для мобильных и встроенных приложений, где важны как точность, так и эффективность вычислений.

E. ResNet

ResNet, или Residual Network, представляет собой инновационную в свое время архитектуру сверточной нейронной сети, предложенную в статье "Deep Residual Learning for Image Recognition" исследователями Microsoft Research в 2015 году. ResNet был представлен с целью преодоления проблемы затухающего градиента и улучшения обучения глубоких нейронных сетей.

Вот основные характеристики ResNet [15, 16].

1) Residual Blocks: основная идея ResNet заключается в использовании блоков с остаточным соединением. Вместо того чтобы строить нейронную сеть, предсказывающую прямое отображение, блоки ResNet предсказывают разницу между входом и выходом. Это достигается добавлением остаточного (или сокращенно "residual") соединения, которое пропускает входные данные вдоль сети. Это позволяет легче обучать глубокие сети, поскольку остаточные соединения упрощают передачу градиента и снижают вероятность затухания градиента.

2) Deep Network Architecture: ResNet может состоять из нескольких слоев, исходно варьируя от десятков до сотен слоев. Это глубокие архитектуры позволяют модели извлекать более сложные и абстрактные признаки из изображений, улучшая их способность к классификации и другим задачам компьютерного зрения.

3) Global Average Pooling: вместо использования полносвязных слоев в конце сети, ResNet обычно применяет глобальный средний пулинг (Global Average Pooling). Это позволяет уменьшить количество параметров и снизить переобучение.

4) Skip Connections: остаточные соединения в блоках ResNet создают "skip connections", или "shortcut connections", которые обеспечивают прямой путь для градиентов вдоль сети. Это позволяет эффективнее передавать градиенты через сеть, содействуя обучению глубоких моделей.

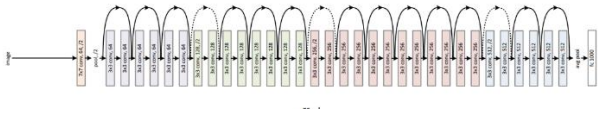


Рис. 5 – Архитектура ResNet50

ResNet часто применяется в различных задачах компьютерного зрения, таких как классификация изображений, обнаружение и сегментация объектов. Его успешное использование подтверждается выдающейся способностью обучаться глубоким моделям, делая его одним из важных вкладов в развитие глубокого обучения.

F. VGG

VGG (Visual Geometry Group) – это группа моделей глубокого обучения, представленных в статье "Very Deep Convolutional Networks for Large-Scale Image Recognition", разработанной исследователями из Visual Geometry Group при Университете Оксфорд в 2014 году. VGG стало известным благодаря своей простой и единообразной архитектуре, которая стала основой для многих последующих исследований в области глубокого обучения.

Вот несколько ключевых аспектов архитектуры VGG [17, 18].

1) Основная архитектура: VGG использует серию сверточных слоев с небольшими фильтрами размером 3x3, и каждый слой следует за другим без использования сложных модулей или переходов. Такой подход обеспечивает глубокую структуру сети.

2) Структура блоков: основной строительный блок VGG состоит из нескольких последовательных сверточных слоев с активацией ReLU, за которыми следует пулинг для уменьшения размера изображения. Такие блоки повторяются несколько раз, создавая глубокую иерархию.

3) Глубина архитектуры: VGG имеет различные версии с разным количеством слоев (например, VGG16 и VGG19). Например, VGG16 состоит из 16 сверточных и полносвязных слоев, включая три полносвязных слоя для классификации.

4) Fully Connected Layers: после сверточных блоков VGG включает несколько полносвязных слоев для окончательной классификации. Эти слои располагаются в конце сети и обычно сопровождаются функцией активации softmax для прогнозирования вероятностей классов.

5) Размер фильтров: VGG использует фильтры размером 3x3 в своих сверточных слоях. Такой размер фильтров обеспечивает приемлемое покрытие изображения и сохраняет вычислительную эффективность.

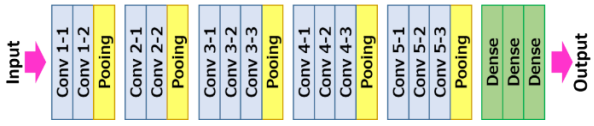


Рис. 6 – Архитектура VGG16

VGG была разработана в первую очередь для решения задачи классификации изображений. Она успешно применяется в соревнованиях по распознаванию изображений, таких как ImageNet Large Scale Visual Recognition Challenge (ILSVRC).

Хотя VGG обладает высокой точностью, ее глубокая структура требует большого количества параметров, что делает ее более затратной с точки зрения вычислительных ресурсов.

IV Построение моделей для задачи классификации транспортных средств

В рамках данной статьи рассмотрим, как две предопученные сверточные нейронные сети, а именно ResNet50 и VGG16, справятся с задачей классификации транспортных средств.

A. Датасет

Модели сначала обучаются на стартовом датасете [14], состоящем из 100 изображений в каждой из четырех категорий: автомобили (car), автобусы (bus), грузовики (truck) и мотоциклы (motorcycle) [19].

После завершения обучения модели оцениваются на новом датасете [20] с аналогичной структурой, содержащем другие изображения в тех же четырех категориях, который был собран из источника.

Этот подход позволяет оценить обобщающую способность моделей и их эффективность в распознавании транспортных средств на разнообразных изображениях.

B. ResNet50

В начале кода определяются параметры, такие как размер изображения (224x224) и размер батча (batch size = 32). Используется ImageDataGenerator для изменение масштаба изображения (1./255) и установки уровня разделения данных на тренировочные и валидационные (0,25). Обучающий и валидационный генераторы создаются с использованием flow_from_directory, где данные разделены на тренировочный и валидационный наборы.

Загружается предварительно обученная модель ResNet50, и ее часть (все слои до последних пяти) замораживается, чтобы сохранить предварительно обученные веса. Затем добавляются новые слои поверх ResNet50: слой выравнивания (Flatten), полносвязный слой с 256 нейронами и функцией активации ReLU, и выходной слой с 4 нейронами и функцией активации softmax для классификации.

После этого модель компилируется с использованием функции потерь categorical_crossentropy, оптимизатора RMSprop с заданным learning rate и метрикой accuracy.

Затем модель обучается модель обучается на тренировочных данных в течение 30 эпох.

Функция потерь на валидационной выборке составила 0,81, а accuracy – 0,69.

На новом датасете функция потерь составила 1,13, а accuracy – 0,45.

Матрица ошибок выглядит следующим образом.

ТАБЛИЦА I – Матрица ошибок нейронной сети ResNet50 на новом датасете

	Bus	**Car**	**Motorcycle**	**Truck**
Bus	10	11	29	50
Car	14	7	17	62
Motorcycle	6	6	24	64
Truck	7	8	26	59

Для класса "Bus" было сделано 10 правильных предсказаний, 11 ошибочных предсказаний как "Car", 29 как "Motorcycle", 50 как "Truck".

Для класса "Car" было сделано 7 правильных предсказаний, 14 ошибочных предсказаний как "Bus", 17 как "Motorcycle", 62 как "Truck".

Для класса "Motorcycle" было сделано 24 правильных предсказаний, 6 ошибочных предсказаний как "Bus", 6 как "Car", 64 как "Truck".

Для класса "Truck" было сделано 59 правильных предсказаний, 7 ошибочных предсказаний как "Bus", 8 как "Car", 26 как "Motorcycle".

Также были выведены 6 изображений с помощью библиотеки Matplotlib для демонстрации работы модели.

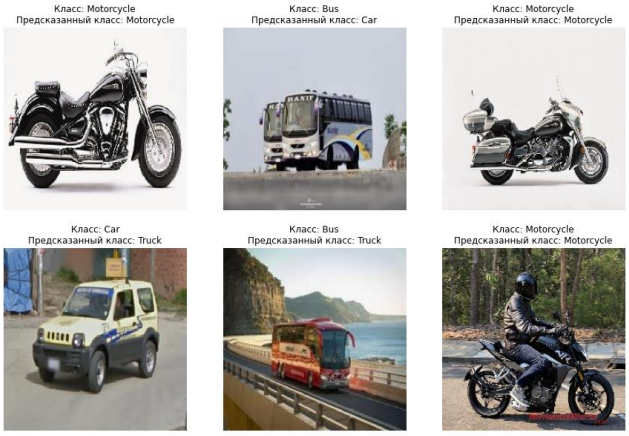


Рис. 7 – Пример классификации транспорта моделью ResNet50

Модель верно предсказала 3 изображения с мотоциклами, однако ошибочно классифицировала автобус как машину, другой автобус как грузовик, машину как грузовик.

C. VGG16

В начале кода определяются параметры, такие как размер изображения (224x224) и размер батча (batch size = 32). Используется ImageDataGenerator для изменение масштаба изображения (1./255) и установки уровня разделения данных на тренировочные и валидационные (0,25). Обучающий и валидационный генераторы создаются с использованием flow_from_directory, где данные разделены на тренировочный и валидационный наборы.

Загружается предварительно обученная модель VGG16, и ее часть (все слои до предпоследних пяти) замораживается, чтобы сохранить предварительно обученные веса. Затем добавляются новые слои поверх VGG16: слой выравнивания (Flatten), полносвязный слой с 256 нейронами и функцией активации ReLU, и выходной

слой с 4 нейронами и функцией активации softmax для классификации.

После этого модель компилируется с использованием функции потерь categorical_crossentropy, оптимизатора RMSprop с заданным learning rate и метрикой accuracy.

Затем модель обучается модель обучается на тренировочных данных в течение 30 эпох.

Функция потерь на валидационной выборке составила 0,21, а accuracy – 0,91.

На новом датасете функция потерь составила 1,47, а accuracy – 0,65.

Матрица ошибок выглядит следующим образом.

ТАБЛИЦА II – МАТРИЦА ОШИБОК НЕЙРОННОЙ СЕТИ VGG16 НА НОВОМ ДАТАСЕТЕ

	Bus	**Car**	**Motorcycle**	**Truck**
Bus	21	7	19	53
Car	15	9	27	49
Motorcycle	28	1	24	47
Truck	20	6	28	46

Для класса "Bus" было сделано 21 правильных предсказаний, 7 ошибочных предсказаний как "Car", 19 как "Motorcycle", 53 как "Truck".

Для класса "Car" было сделано 9 правильных предсказаний, 15 ошибочных предсказаний как "Bus", 27 как "Motorcycle", 49 как "Truck".

Для класса "Motorcycle" было сделано 24 правильных предсказаний, 28 ошибочных предсказаний как "Bus", 1 как "Car", 47 как "Truck".

Для класса "Truck" было сделано 46 правильных предсказаний, 20 ошибочных предсказаний как "Bus", 6 как "Car", 28 как "Motorcycle".

Также были выведены 6 изображений с помощью библиотеки Matplotlib для демонстрации работы модели.

Рис. 8 – Пример классификация транспорта моделью VGG16

Модель верно предсказала 3 изображения: автобус, грузовик и мотоцикл. Ошиблась также в 3 изображениях: автобус предсказала как грузовик, машину предсказала как грузовик, грузовик предсказала как автобус.

D. Сравнение полученных моделей

VGG16 показывает более низкую функцию потерь и более высокую точность на валидационной выборке: 0,21 и 0,91 у VGG16 против 0,81 и 0,69 у ResNet50 соответственно.

На новом датасете VGG16 вновь показываем более высокую точность, однако и более высокую функцию потерь: 1,47 и 0,65 у VGG16 против 0,13 и 0,45 у ResNet50 соответсвенно.

Из матриц ошибок видно, что у обеих моделей присутствует ряд ошибочных классификаций. Для модели ResNet50 распространена ошибочная классификация автобусов как грузовиков и машин, для VGG16 – автобусов как грузовиков и машин, а также грузовиков как автобусов.

У обеих моделей присутствует проблема с классификацией грузовиков, однако VGG16 показывает более высокую точность классификации.

Таким образом, модель VGG16 выглядит предпочтительней для классификации транспортных средств, хоть и нуждается в доработке.

V ВЫВОДЫ

В данной статье были рассмотрены ключевые сверточные нейронные сетей, их строение, проблемы затухающего градиента и инновационные методы, используемые для их преодоления. Также были изучены различные архитектуры предобученых сверточных нейронных сетей, такие как AlexNet, DenseNetm, GoogLeNet, MobileNet, ResNet и VGG.

Затем были были проанализированны две важные архитектуры сверточных нейронных сетей, ResNet50 и VGG16, в контексте их применения к задаче классификации транспортных средств.

Построение моделей ResNet50 и VGG16 для классификации транспортных средств было реализовано с использованием предварительно обученных моделей и дополнительных слоев для адаптации к конкретной задаче. Обучение проводилось на стартовом датасете, а оценка моделей проводилась на новом датасете, что позволило оценить их обобщающую способность.

В результате экспериментов стало ясно, что VGG16 демонстрирует более высокую точность и более низкую функцию потерь на валидационной выборке по сравнению с ResNet50. Однако обе модели допускают ошибки, особенно в классификации автобусов и грузовиков.

Дополнительные шаги для улучшения результатов могут включать в себя тонкую настройку гиперпараметров, использование других предобученных моделей или увеличение размера и разнообразия датасета.

СПИСОК ЛИТЕРАТУРЫ

[1] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[2] li, Bushra & Sadekov, Rinat. (2023). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation. 30. 87–105. 10.17285/0869-7035.00105.

[3] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[4] Tanchenko, A. & Fedulin, A. & Bikmaev, R. & Sadekov, Rinat. (2020). UAV Navigation System Autonomous Correction Algorithm Based on Road and River Network Recognition. Gyroscopy and Navigation. 11. 293-299. 10.1134/S2075108720040100.

[5] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[6] Как работает сверточная нейронная сеть (CNN). [Электронный ресурс]. – Ресурс доступа: https://neurohive.io/ru/osnovy-data-science/glubokaya-svertochnaja-nejronnaja-set// (дата обращения: 06.01.2024).

[7] AlexNet — свёрточная нейронная сеть для классификации изображений. [Электронный ресурс]. – Ресурс доступа: https://neurohive.io/ru/vidy-nejrosetej/alexnet-svjortochnaja-nejronnaja-set-dlja-raspoznavanija-izobrazhenij/ (дата обращения: 06.01.2024).

[8] Minhas RA, Javed A, Irtaza A, Mahmood MT, Joo YB. Shot Classification of Field Sports Videos Using AlexNet Convolutional Neural Network. Applied Sciences. 2019; 9(3):483.

[9] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708

[10] DenseNet Architecture Explained with PyTorch Implementation from TorchVision. [Электронный ресурс]. – Ресурс доступа: https://amaarora.github.io/posts/2020-08-02-densenets.html (дата обращения: 06.01.2024).

[11] Understanding GoogLeNet Model – CNN Architecture. [Электронный ресурс]. – Ресурс доступа: https://www.geeksforgeeks.org/understanding-googlenet-model-cnn-architecture/ (дата обращения: 06.01.2024).

[12] GoogleNet Architecture Implementation in Keras with CIFAR-10 Dataset. [Электронный ресурс]. – Ресурс доступа: https://machinelearningknowledge.ai/googlenet-architecture-implementation-in-keras-with-cifar-10-dataset/ (дата обращения: 06.01.2024).

[13] Howard, Andrew & Zhu, Menglong & Chen, Bo & Kalenichenko, Dmitry & Wang, Weijun & Weyand, Tobias & Andreetto, Marco & Adam, Hartwig. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications.

[14] Классификация изображений с помощью MobileNet. [Электронный ресурс]. – Ресурс доступа: https://skine.ru/articles/222907/ (дата обращения: 06.01.2024).

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770-778

[16] Residual Networks (ResNet) – Deep Learning. [Электронный ресурс]. – Ресурс доступа: https://www.geeksforgeeks.org/residual-networks-resnet-deep-learning/ (дата обращения: 06.01.2024).

[17] Simonyan, Karen & Zisserman, Andrew. (2014). Very Deep Convolutional Networks for Large-Scale Image Recognition.

[18] VGG Very Deep Convolutional Networks (VGGNet) – What you need to know. [Электронный ресурс]. – Ресурс доступа: https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/ (дата обращения: 06.01.2024).

[19] Vehicle Type Recognition. [Электронный ресурс]. – Ресурс доступа: https://www.kaggle.com/datasets/kaggleashwin/vehicle-type-recognition (дата обращения: 15.12.2023).

[20] Roboflow. [Электронный ресурс]. – Ресурс доступа: https://roboflow.com/ (дата обращения: 17.12.2023).

Применение Instant NeRFs для создания трехмерных изображений

Д. В. Савенков
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m2312188@edu.misis.ru

Д. В. Лоткова
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1908659@edu.misis.ru

***Аннотация* — данная научная статья посвящена исследованию и применению технологии NVIDIA Instant NeRFs (Neural Radiance Fields) в области создания трехмерных изображений. Методы генерации трехмерных сцен становятся все более востребованными в виртуальной реальности, компьютерной графике и других областях. В статье рассматривается эффективность и универсальность использования алгоритма Instant NeRFs, основанного на глубоком обучении, для создания фотореалистичных трехмерных моделей. В данной статье представляется обзор архитектуры Instant NeRFs, выявляются его преимущества и оцениваются результаты экспериментов на различных наборах данных. Полученные результаты подчеркивают потенциал данного метода в контексте трехмерного моделирования и визуализации при помощи воксельной графики, делая его значимым вкладом в область компьютерной графики и искусственного интеллекта.**

Ключевые слова — нейронные сети, 3D моделирование, 3D реконструкция, синтез изображений, изображения, виртуальная реальность, Neural Radiance Fields.

I. Введение

Искусственные нейронные сети представляют собой вычислительные системы, инспирированные биологической организацией нервной системы живых существ. Они используют архитектуру, состоящую из искусственных нейронов, объединенных в слои, для обработки информации. Эти сети обучаются на основе данных, чтобы распознавать шаблоны, делать прогнозы или выполнять задачи, для которых им необходим опыт. Искусственные нейронные сети находят широкое применение в различных областях, таких как системы навигации [1], распознавание образов [2], обработка естественного языка [3], компьютерное зрение [4] и многие другие. Например, в задачах классификации изображений глубокие нейронные сети способны автоматически распознавать и классифицировать объекты на фотографиях. Искусственные нейронные сети также эффективно применяются в задачах прогнозирования, управления процессами и в робототехнике [5], что подчеркивает их универсальность и эффективность в различных областях применения.

В последние десятилетия нейронные сети стали ключевым инструментом в области компьютерного зрения, позволяя решать широкий спектр задач, включая преобразование изображений 2D в трехмерные объекты. Этот процесс, известный как преобразование изображения 2D в 3D, представляет собой значительный вызов, требующий точности и высокой степени абстракции [6].

NeRF (Neural Radiance Field) — технология, разработанная командой NVIDIA и поддерживаемая сложной нейронной сетью, оптимизированной с помощью фирменного кода под названием Instant NeRF. Эта технология позволяет быстро и точно создавать детализированные 3D-модели физических сред, используя гораздо меньше фотографических данных, чем альтернативные методы фотомоделирования, которые напрямую не поддерживают искусственный интеллект [7]. Neural radiance field (NeRF), в частности его расширение с помощью примитивов мгновенной нейронной графики, представляет собой новый метод рендеринга для синтеза представлений, который использует изображения реального мира для создания фотореалистичных иммерсивных виртуальных сцен [8].

Исследования в области неявных нейронных представлений (INR) в последние годы стали одним из наиболее перспективных направлений исследований в области компьютерного зрения. Это связано с тем, что INR обладают свойством сквозной дифференцируемости, что позволяет использовать их для решения широкого круга задач, включая передачу стиля и редактирование дифференцируемых форм [9]. В отличие от традиционных подходов, которые представляют объекты в виде дискретной сетки, INR представляют объекты в виде непрерывной функции. Важной особенностью является их независимость от разрешения, поскольку они могут быть запрошены в произвольных точках входных данных, в отличие от кодирования информации в фиксированной сетке или последовательности.

NeRF представляет собой INR, описывающее 3D-сцену в виде функции, сопоставляющей координаты и направления просмотра с плотностями и цветами RGB. Суть состоит в том, что мы представляем трехмерную поверхность с помощью описания формы и radiance [10]. Мы задаем не просто текстуру, а свет, который отражается от этой точки в определенном направлении, таким образом, учитывая тип поверхности. Используя такую модель, мы пытаемся тренировать ее по изображениям [11]. Для представления формы поверхности используется модель, которая называется «объемная плотность», которая предсказывает вероятность того, что луч оборвется именно в этой точке: если 1 – значит, есть объект, если 0 – значит, пустое пространство.

Таким образом, мы можем считать цвет конкретного пикселя, зная внутренние параметры камеры, выпуская луч и считая интеграл, который зависит от объемной плотности вдоль этого луча и от цвета в radiance модели в каждой из этих точек. Этот метод позволяет создавать

реалистичные изображения 3D-сцен, что делает их мощным инструментом для компьютерного зрения.

II. Наборы данных

Пайплайн принимает как фото, так и видео входные данные для мгновенного создания NeRF. Первый этап в пайплайне создания NeRF использует COLMAP для определения положений камер [12]. В связи с этим необходимо следовать основным принципам фотограмметрии в отношении перекрытия и четкости изображений.

Для обучения и тестирования нейронной сети, рассматриваемой в данной работе, использовались два наборы данных, как локальные, собранные авторами, так и открытые. Рассмотрим используемый открытый набор.

A. Открытый набор данных

Рис. 1. Примеры фото из открытого набора данных «LEGO»

Данный датасет был предложен компанией NVIDIA для демонстрации возможностей нейронной сети. Он состоит из 102 фотографий, разрешение: 1015x764p.

Реализация NeRF по умолчанию осуществляет лучи через ограничивающий параллелепипед с единичными размерами, от [0, 0, 0] до [1, 1, 1]. Загрузчик данных по умолчанию берет трансформации камер из входного файла JSON, масштабирует положения на 0.33 и смещает на [0.5, 0.5, 0.5], чтобы сопоставить начало входных данных с центром этого куба [13]. Фактор масштабирования выбран таким образом, чтобы соответствовать синтетическим наборам данных в оригинальной работе по NeRF, а также результатам осуществляемого сценария.

B. Локальный набор данных

Процесс обучения довольно чувствителен к качеству набора данных. Например, важно, чтобы у набора данных было хорошее покрытие, отсутствие неверных меток камеры и отсутствие размытых кадров (проблематичны как движущиеся, так и дефокусировочные размытия) [14].

В данном примере предполагается использование одного видеофайла в качестве входных данных, после чего извлекаются кадры с заданной частотой кадров.

Данный датасет состоит из 35-секундного видео, получилось 351 фотография с разрешением 1080x1920p, параметр aabb_scale равен 2, а параметр video_fps 10.

Рис. 2. Кадр из локального набора данных «TEAPOT»

Стоит упомянуть, что параметр aabb_scale является ключевым и имеет наибольшее значение. Он определяет масштаб сцены, по умолчанию установленный на 1. Это означает, что сцена масштабируется так, чтобы положения камер находились на среднем расстоянии 1 единицы от исходной точки. Для маленьких синтетических сцен, таких как оригинальные данные NeRF, значение по умолчанию aabb_scale (равное 1) идеально подходит и обеспечивает быстрое обучение.

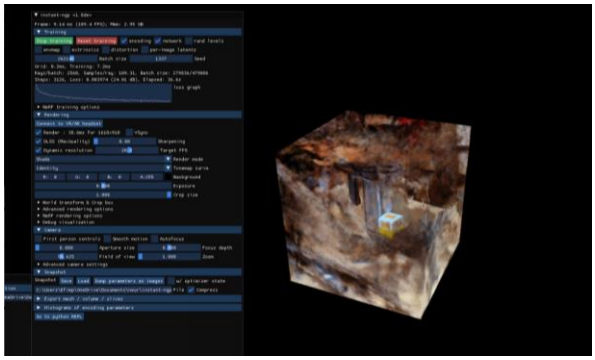


Рис. 3. Внешний вид сцены 3D изображения из локального набора данных «TEAPOT» с параметром aabb_scale = 1

Модель NeRF предполагает, что обучающие изображения могут быть полностью описаны сценой в пределах этой ограничивающей рамки. Однако для естественных сцен, где фон выходит за границы этой рамки, модель может столкнуться с трудностями и вызвать галлюцинации "плавающих объектов" на границах рамки.

Путем установки более высокого значения aabb_scale, равного 8 (максимум 128), модель NeRF расширяет лучи до значительно большей ограничивающей рамки. Важно отметить, что это может оказать небольшое влияние на скорость обучения.

III. Нейросетевая архитектура

Архитектура Neural Radiance Fields (NeRF) представляет собой подход к воссозданию трехмерных сцен и их освещения с использованием нейронных сетей. Следует подчеркнуть, что каждую сеть необходимо обучить для захвата конкретной сцены. В отличие от стандартного машинного обучения, основная цель заключается в переобучении нейронной сети на определенную сцену.

Фактически нейронные поля внедряют сцену в параметры весов нейронной сети [15].

NeRF принимают в качестве входных данных одну непрерывную 5D-координату, которая состоит из пространственного положения (x, y, x) и направления обзора (θ , ϕ). Эта конкретная точка объекта/сцены подается на вход многослойному персептрону (MLP), который выдает соответствующие цветовые интенсивности c = (r, g, b) и плотность объема σ. Плотность объема (вероятности) указывает, сколько светимости (или яркости) накапливается лучом, проходящим через точку (x, y, z), и представляет собой меру "воздействия" этой точки на общую сцену [16]. Интуитивно, плотность объема вероятности предоставляет вероятность того, что предсказанное значение цвета следует учитывать.

Задача обучения таких архитектур заключается в том, что целевые значения плотности и цвета неизвестны. Поэтому нам требуется (дифференцируемый) метод, чтобы отобразить их обратно на двумерные изображения. Затем эти изображения сравниваются с эталонными изображениями, формулируя потери визуализации, по которым мы можем оптимизировать сеть. (Рисунок 3)

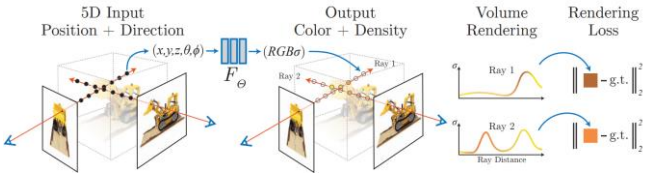


Рис. 4. Схема тренировочного процесса NeRFs

Как видно на представленном изображении, для преобразования выходных данных нейронного поля в двумерное изображение используется объемная визуализация [17]. NeRF использует ускоренный метод трассировки лучей при обучении и рендеринге. В процессе перемещения по лучу образцы распределяются так, чтобы они равномерно вносили вклад в изображение, минимизируя потери вычислений, а также излишние вычислительные ресурсы.

NVIDIA Instant NeRF (Neural Radiance Fields) представляет собой глубокую нейронную сеть, используемую для воссоздания трехмерных сцен на основе входных данных в виде множества изображений с разных точек обзора. Авторы применяют графические нейронные примитивы, в сочетании с оригинальным представлением входных данных, называемым многоразрядным хэш-кодированием. Этот вид кодирования позволяет использовать небольшие нейронные сети, что снижает общее количество операций с плавающей точкой. Кроме того, авторы предлагают использовать специальные реализации на графическом процессоре для каждой задачи, что еще более снижает общую вычислительную сложность. Одним из таких предложений является реализация всей многослойной нейронной сети в виде единого ядра CUDA, чтобы каждое вычисление выполнялось в локальном кэше графического процессора [18].

Вместо того чтобы обучать только параметры сети, мы также обучаем параметры кодирования (векторы признаков). Эти векторы упорядочиваются по разным уровням разрешения и сохраняются в узлах сетки. В больших сценах также применяется каскадирование сетки занятости и распределение образцов экспоненциально вдоль луча. Применение сетки занятости позволяет сконцентрировать образцы около поверхностей, улучшая качество рендеринга.

Основные компоненты архитектуры включают MLP плотности (Density MLP) и MLP цвета (Color MLP), которые объединяются для создания модели способной репродуцировать сложные трехмерные окружения. MLP плотности (Density MLP) с одним или несколькими скрытыми слоями, на входе закодированная хеш-позиция. В стандартной модели это отображает вход в 16 выходных значений. Первое из этих значений интерпретируется как логарифм плотности. MLP цвета (Color MLP) принимает на вход 16 выходных значений MLP плотности и вида направления. Выход представляет собой RGB цветовую тройку. В зависимости от динамического диапазона данных обучения используется сигмоидная активация (sRGB) или экспоненциальная активация (линейный HDR) [19].

Весь процесс (Рисунок 4) полностью дифференцируем. Для обучения кодировок градиенты потерь распространяются через MLP, конкатенацию и линейную интерполяцию, а затем накапливаются в векторах признаков, полученных из таблиц поиска. Также важно отметить, что этот подход полностью независим от конкретной задачи и может быть использован для различных архитектур и задач, помимо NeRFs.

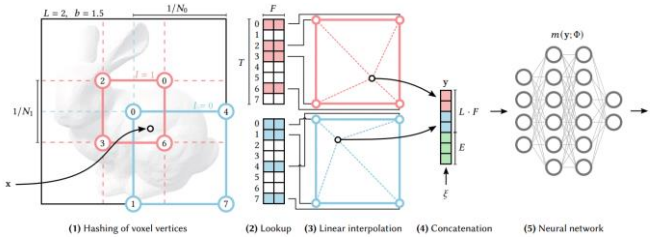


Рис. 5. Хэш-кодирование: 1) Определение окружающих сеток и присваивание индексов вершинам, хеширование их координаты; 2) Обращение к соответствующим обучаемым векторам признаков; 3) Линейная интерполяция для объединения векторов признаков; 4) Конкатенация вектора с другими вспомогательными входами; 5) Полученный вектор признаков подается на вход нейронной сети.

Обучая параметры кодирования параллельно с нейросетью, мы получаем значительный прирост качества конечного результата. Используя несколько разрешений, достигается автоматический уровень детализации, что означает, что сеть обучается как грубым, так и тонким особенностям. При использовании хэширования для связи трехмерного пространства с векторами признаков процесс кодирования становится полностью независимым от конкретной задачи.

Эта комплексная архитектура позволяет NeRF эффективно обучаться данных, представленных в виде облака точек, и создавать изображения высокого качества с учетом освещения и детализации. Таким образом, NVIDIA Instant NeRF создает реалистичные трехмерные визуализации сцен, обучаясь на множестве изображений и используя глубокие нейронные сети для представления плотности и цвета объектов в сцене.

IV. Проведение испытаний

Для тестирования работоспособности выбранной нейронной модели сперва её требуется обучить. В данном случае переменная, настройка которой напрямую повлияет на успешность обучения это качество изображений и выбор количества кадров. Технические параметры устройства, на котором производится обучение: процессор – AMD Ryzen 7 3700X 8 – Core Processor, оперативная память – 16 Гб, видеокарта - RTX 3070 Ti.

Для открытого датасета: готовый набор данных в формате transforms.json, должен быть центрирован относительно начала координат и иметь схожий масштаб с оригинальными синтетическими наборами данных NeRF. При загрузке его в NGP, первое, что следует проверить, — это положение камер относительно единичного параллелепипеда, используя отладочные возможности. Если большая часть набора данных не попадает в единичный параллелепипед, имеет смысл переместить его туда. Это можно сделать, отрегулировав сами преобразования, или добавив глобальные параметры во внешний контекст JSON.

В результате, для открытого датасета, состоящего из 102 фотографий с разрешением 1015x764p, обучение заняло 4 минуты.

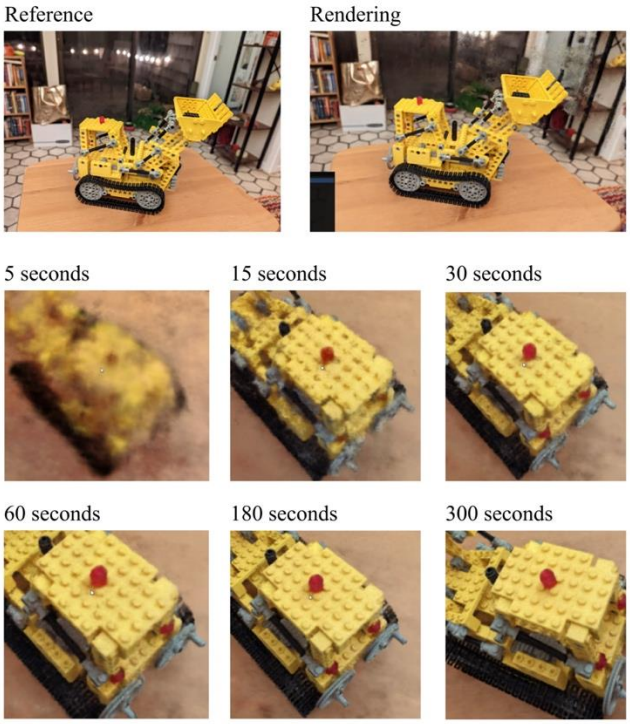


Рис. 6. Результат работы NVIDIA Instant NeRFs для набора данных «LEGO»

Далее было проведено сравнение референсных изображений с оценочными с использованием метода наименьших квадратов (МНК). Сравнение двух кадров проводилось по 3 каналам (RGB), всего было выбрано рандомно 10% от исходного датасета. Для открытого набора данных средний показатель точности изображения равен 99,236%.

ТАБЛИЦА I. Оценка датасета «LEGO» МНК

Кадр	MSE (%)	Кадр	MSE (%)
1	99,93	6	99,52
2	99,23	7	99,26
3	99,49	8	99,74
4	97,70	9	98,83
5	98,82	10	99,84

Для создания следующей 3D сцены подается видеофайл в качестве входных данных, после чего извлекаются кадры с заданной частотой кадров.

В результате, для локального датасета "TEAPOT", состоящего из видеофайла, длительностью 35 секунд, с параметрами: "fps" = 10, aabb_scale = 8, мы получили 351 изображение и затраченное время на извлечение кадров и обучение нейронной сети заняло 17 минут.

Рис. 7. Результат работы NVIDIA Instant NeRFs для набора данных «TEAPOT»

Для локального набора данных средний показатель точности изображения равен 98,576%.

ТАБЛИЦА II. Оценка датасета «TEAPOT» МНК

Кадр	MSE (%)	Кадр	MSE (%)
1	96,85	18	98,89
2	94,93	19	99,45
3	98,98	20	99,56
4	99,34	21	95,87
5	96,72	22	99,92
6	99,23	23	97,78
7	97,42	24	99,76
8	99,54	25	99,43
9	98,72	26	99,24
10	99,12	27	95,43
11	97,85	28	99,56
12	99,47	29	97,74
13	95,63	30	99,89
14	99,92	31	96,92

15	99,67	32	99,78
16	99,54	33	99,92
17	99,75	34	99,78

Поэтапное описание визуализации набора данных:

- 5 секунд: на этом этапе нейросеть проводит первичный анализ визуальных данных, строя приблизительные модели объектов, и определяет их расположение в пространстве. Это может включать в себя выделение основных контуров и общих характеристик объектов.
- 15 секунд: в течение следующего временного интервала нейросеть улучшает свою способность распознавать цвета в модели и дополняет представление, создавая более точные формы объектов. Это может включать в себя анализ цветовых пикселей и более детальное описание форм объектов.
- 30 секунд: нейросеть начинает интегрировать окружение и более мелкие детали в свое представление. Это включает в себя анализ фоновых элементов, текстур и дополнительных объектов, что позволяет модели более полно воссоздать сцену.
- 60 секунд: нейросеть продолжает улучшать свое представление, более детально строя окружение и модель. На этом этапе возможно улучшение разрешения изображения и более точное определение особенностей.
- 180 секунд: время, когда нейросеть становится способной распознавать текст и более четко воспринимать изображение. Это может включать в себя работу с высокочастотными деталями, что позволяет сети определять более мелкие элементы, такие как буквы и мелкие детали текстур.
- 300 секунд: на этом этапе нейросеть полностью интегрирует передний и задний план, достигая полной четкости в представлении изображения. Это включает в себя способность различать надписи и дополнительные детали в окружении.

Этот процесс можно охарактеризовать как постепенное улучшение восприятия нейросети, начиная с общих черт и постепенно переходя к более детальному и точному представлению визуальных данных.

Тренировочные изображения примерно указывают на общую точку интереса, которую они помещают в начало координат. Эта точка находится путем взвешенного усреднения ближайших точек пересечения лучей через центральный пиксель всех пар тренировочных изображений. На практике это означает, что сценарий работает лучше всего, когда тренировочные изображения были захвачены, направлены внутрь к объекту интереса, хотя они не обязательно должны полностью охватывать его в полном 360-градусном обзоре. Любой видимый фон за объектом интереса будет восстановлен, если aabb_scale установлен на число больше 1.

Локальный датасет, содержащий изображения чайника (TEAPOT), продемонстрировал более высокое качество рендеринга, вероятно, благодаря обширному объему включенных изображений (351 шт.) и их удаленности. Эта характеристика, вероятно, способствует лучшему охвату разнообразия форм, освещения и углов обзора, что содействует обучению модели воспроизводить изображения более точно и реалистично.

В отношении датасета Lego, содержащий изображение трактора, который включает в себя меньшее количество изображений (102 шт.), несмотря на это, результаты рендеринга также оказались приемлемыми. Возможно, это объясняется тем, что даже при ограниченном объеме данных датасет предоставил достаточное разнообразие сценариев и условий для обучения модели.

Таким образом, мы наблюдаем, что эффективность рендеринга зависит не только от количества включенных изображений, но и от их характеристик, таких как удаленность и разнообразие. Больший объем данных и широкий спектр сценариев, обогащают обучающий датасет, способствуя улучшенной способности модели к генерации качественных изображений.

V. Заключение

Исследование, представленное в статье, описывает технологию NVIDIA Instant NeRFs (Neural Radiance Fields), который позволяет обучать нейрографические примитивы всего за несколько секунд и визуализировать их с высокой скоростью, превосходящей реальное время. Это достигается благодаря трём ключевым компонентам: новому методу кодирования входных данных для нейросетей, новой структуре CUDA для обучения и вывода, а также эффективным алгоритмам для дифференцируемой визуализации графических примитивов.

Эксперименты показали, что модель NeRF достигает оптимальных результатов при использовании от 50 до 150 изображений, на которых минимизировано движение сцены. Таким образом, для видео длительностью в одну минуту частота кадров (--video_fps) 2 является оптимальным выбором по скорости обучения нейронной сети. Качество восстановления сцены напрямую зависит от точности извлечения параметров камеры из предоставленных изображений.

Также в статье обсуждается важность выравнивания тренировочных данных относительно единичного параллелепипеда и возможность обучения "floaters" кода внешнего вида для каждого изображения, что позволяет улучшить качество реконструкции в случае несогласованности освещения и баланса белого. Выявлено, что при использовании однородного фонового цвета модель может минимизировать свои потери, просто предсказывая этот цвет фона, вместо учета прозрачности (нулевой плотности). Случайное изменение цветов фона заставляет модель обучаться нулевой плотности, что позволяет случайным цветам фона "прорываться" сквозь изображение. И, наконец, в отношении обучаемых кодировок входных данных мы обнаружили, что многоразрешенные хэш-таблицы обладают множеством привлекательных характеристик. Они могут автоматически изучать разреженность способом, независящим от задачи, как простой побочный продукт использования градиентного спуска, и, таким образом, могут быть использованы как общий строительный элемент для графических примитивов нейронов.

В целом, описанный метод демонстрирует значительное ускорение обучения и визуализации графических

примитивов, открывая новые перспективы в области компьютерной графики и визуальных вычислений.

Литература

[1] D. B. Pazychev and R. N. Sadekov, "Simulation of INS Errors of Various Accuracy Classes," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-3

[2] Баканов П.П., Измайлов Л.С., Тригуб Н.А. ФОРМИРОВАНИЕ ЧИСЛОВОГО КОДА ФРАКТАЛЬНОЙ СТРУКТУРЫ ТЕКСТУРИРОВАННОГО ОПТИЧЕСКИ АНИЗОТРОПНОГО ГЛАСТЭЛИТА // Перспективы науки . - 2023. - №5. - С. 118-125.

[3] Berdichevskaia A. Atypical lexical abbreviations identification in Russian medical texts //2022 12th International Conference on Pattern Recognition Systems (ICPRS). – IEEE, 2022. – С. 1-5.

[4] R. R. Bikmaev, M. D. Zolotov, A. N. Popov and R. N. Sadekov, "Improving the Accuracy of Supporting Mobile Objects with the Use of the Algorithm of Complex Processing of Signals with a Monocular Camera and LiDAR," 2019 26th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2019, pp. 1-4, doi: 10.23919/ICINS.2019.8769360.

[5] Практическое применение роботов и сопутствующих технологий в борьбе с пандемией COVID-19 / А. Р. Ефимов, А. С. Гонноченко, Д. Б. Пайсон [и др.] // Робототехника и техническая кибернетика. – 2020. – Т. 8, № 2. – С. 87-100.

[6] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N.Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in anAutonomous Tram to Predict the Behaviour of Vehicles in the RoadScene," 2022 29th Saint Petersburg International Conference onIntegrated Navigation Systems (ICINS), Saint Petersburg, RussianFederation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388

[7] Jonathan Stephens. Getting Started with NVIDIA Instant NeRFs / URL: https://developer.nvidia.com/blog/getting-started-with-nvidia-instant-nerfs/ (дата обращения: 10.12.23)

[8] (PDF) Instant Neural Graphics Primitives with a Multiresolution Hash Encoding from: https://nvlabs.github.io/instant-ngp/assets/mueller2022instant.pdf [accessed Dec 28 2023].

[9] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal,P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S.,Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A.,Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E.,Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S.,Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Advances in Neural Information Processing Systems(NeurIPS), 2020.

[10] Neural Fields in Visual Computing and Beyond / Yiheng Xie и др. // https://arxiv.org/pdf/2111.11426.pdf - 2022.

[11] NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis / Ben Mildenhal и др. // https://arxiv.org/pdf/2003.08934.pdf - 2020

[12] Tips for training NeRF models with Instant Neural Graphics Primitives, 2023 / URL: https://github.com/NVlabs/instant-ngp/ (дата обращения: 11.12.23)

[13] Thomas Müller. Turn 2D Images into Immersive 3D Scenes with NVIDIA Instant NeRF in VR, 2023 / URL: https://developer.nvidia.com/blog/turn-2d-images-into-immersive-3d-scenes-with-nvidia-instant-nerf-in-vr/ (дата обращения: 11.12.23)

[14] (PDF) NERF++: ANALYZING AND IMPROVING NEURAL RADIANCE FIELDS / Kai Zhang и др. // https://arxiv.org/pdf/2010.07492.pdf - 2020

[15] Neural Fields in Visual Computing / Towaki Takikawa, Shunsuke Saito и др. // https://arxiv.org/abs/2111.11426v1 - 2021

[16] Sergios Karagiannakos. How Neural Radiance Fields (NeRF) and Instant Neural Graphics Primitives work // URL: https://theaisummer.com/nerf/ (дата обращения: 19.12.23)

[17] NEURAL VOLUME RENDERING: NERF AND BEYOND / Frank Dellaert, Lin Yen-Chen // arXiv:2101.05204v2 [cs.CV] - 2021

[18] Compressible-composable NeRF via Rank-residual Decomposition / Jiaxiang Tang и др. // arXiv:2205.14870v2 [cs.CV] - 11 Oct 2022

[19] Instant Neural Graphics Primitives with a Multiresolution Hash Encoding / THOMAS MÜLLER и др. // arXiv:2201.05989v2 [cs.CV] - 4 May 2022

Исследование возможности классификации дорожных знаков

Карякин А. В.
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2312716@edu.misis.ru

***Аннотация* — В данной статье проводится сравнительный анализ эффективности двух нейросетевых архитектур - ResNet50 и AlexNet - для задачи классификации дорожных знаков. Для экспериментов использовался набор данных GTSRB, содержащий изображения дорожных знаков Германии с различными вариациями. Предобученные модели ResNet50 и AlexNet были дообучены на части данных GTSRB. Результаты показали небольшое преимущество ResNet50, однако AlexNet оказалась более эффективной по времени обучения. Таким образом, в контексте данной задачи предпочтительнее использовать AlexNet, несмотря на чуть худшие метрики. В целом обе модели продемонстрировали сходные результаты по классификации дорожных знаков.**

Ключевые слова — классификация, дорожные знаки, компьютерное зрение, нейронные сети, AlexNet, ResNet50, GTSRB.

I. Введение

В современном мире наблюдается значительный рост использования нейронных сетей в самых различных сферах деятельности. Например, нейронные сети используются для оценки точности позиционирования трамваев в городских условиях [1]. Исследование [2] демонстрирует применение нейронных сетей для прогнозирования поведения транспортных средств. Применение этой технологии можно также увидеть в разработке алгоритмов для беспилотных летательных аппаратов [3][4]. Также нейронные сети имеют широкое применение в медицине, например, при использовании компьютерной томографии [5]. Искусственный интеллект обладает значительным потенциалом в научных исследованиях, так как позволяет анализировать скрытые от человека закономерности [6].

Дорожные знаки — критически важный элемент безопасности и организации дорожного движения, который несет в себе различную информацию для участников движения. Классификация дорожных знаков наряду с детектированием являются важными задачами и имеют множество практических применений: автоматическое управление транспортными средствами, навигация, анализ дорожной ситуации и т. д.

Однако классификация дорожных знаков представляет собой сложную задачу, которая требует учета различных факторов, таких как разнообразие форм, цветов, символов и текстов на знаках, а также влияние условий освещения, погоды, загрязнения, повреждений и перекрытий знаков другими объектами. Кроме того, существует множество национальных и региональных систем дорожных знаков, которые имеют свои особенности и отличия.

В связи с этим, в настоящей статье исследуются несколько подходов к классификации дорожных знаков, основанных на свёрточных нейросетевых архитектурах, таких как ResNet50, AlexNet.

Целью исследования является сравнение эффективности нейросетевых архитектур в контексте классификации дорожных знаков, а также выявление наиболее успешного решения. Результаты данного исследования могут иметь практическое значение, улучшая точность и надежность систем, отвечающих за безопасность и эффективность дорожного движения.

II. Анализ работ

Мринал Халой в своей работе [7] представляет модифицированную Inception сеть со слоями пространственного преобразования для классификации дорожных знаков. Эти слои сделали сеть более устойчивой к деформациям и в результате авторами была достигнута точность в 99.81% на наборе данных GTRSB.

В исследовании Амара Динеш Кумар [8] для классификации дорожных знаков была использована капсульная нейронная сеть. Эти сети известны тем, что хорошо справляются с изменением позы и ориентации объектов и авторами была достигнута точность в 97.6% на наборе данных GTSRB.

Работа Доганчан Темель [9] проводилось исследование влияния неблагоприятных условий на распознавание дорожных знаков. Авторы создали набор данных CURE-TSD-Real с 12 типами сложных условий и показали, что при серьезных условиях средняя точность падает на 29%, полнота - на 68%. Также ими был проведен спектральный анализ условий и показана корреляция между характеристиками спектра и производительностью.

III. Набор данных

GTSRB представляет из себя набор данных, состоящий из изображений дорожных знаков, которые были собраны в Германии [10]. Этот датасет был представлен в рамках соревнования, проведенного в 2011 году, и с тех пор используется для разработки и оценки алгоритмов распознавания дорожных знаков.

Набор данных содержит более чем 50 тысяч изображений, каждому из которых присвоен один из 43 классов. Изображения имеют различные вариации по освещению, положению, углам обзора и т. д.

Рисунок 1. Примеры изображений из GTSRB

Также из особенностей данных можно выделить, что размер изображений не статичен. Изображения имеют размеры начиная с 15x15 пикселей до 250x250 пикселей. Кроме того, изображения хранятся в формате PPM, что требует некоторой обработки, так как не все фреймворки позволяют работать с данным форматом. Также стоит отметить, что дорожные знаки не всегда расположены в центре изображения.

IV. Нейросетевые архитектуры

A. *ResNet50*

ResNet50 – архитектура свёрточной нейронной сети, разработанная исследователями из Microsoft Research и представленная на конференции CVPR в 2015 году [11].

Модель состоит из 50 слоёв. После каждой свёртки используется батч-нормализация, делая обучение более стабильным и быстрым. В качестве функции активации используются ReLU (Rectified Linear Unit):

$$f(x) = max(0, x) \quad (1)$$

После начального слоя следует Max Pooling слой, который дополнительно уменьшает размерность данных. В конце модели используется Global Average Pooling, что помогает уменьшить количество параметров и предотвратить переобучение.

На рисунке 2 представлена архитектура модели:

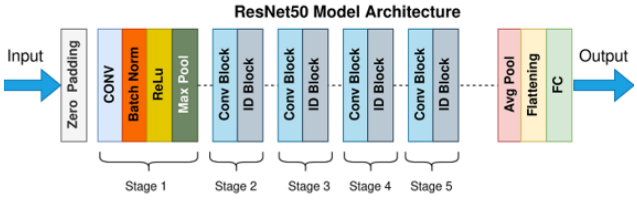


Рисунок 2. Архитектура сети ResNet50

ResNet50 является одной из наиболее популярных архитектур в области классификации изображений, обнаружения объектов и сегментации.

В данной работе использовалась предобученная сеть ResNet50 из torchvision.models. Последний слой был заменён для предсказания одного из 43 классов. Модель была дообучена. Предварительно были заморожены веса всех слоёв, кроме последнего. В качестве функции потерь использовался CrossEntropyLoss, а как оптимизатор был выбран Adam с learning rate равным 0.001. Процесс обучения состоял из 10 эпох.

B. *AlexNet*

Архитектура AlexNet была разработана Алексеем Крижевским совместно с Ильей Суцкевером и Джеффри Хинтоном в 2012 году [12]. Эта нейронная сеть выиграла соревнование ImageNet Large Scale Visual Recognition в 2012 году, достигнув очень высокой, рекордной для того времени точности в задаче классификации изображений на наборе данных ImageNet.

Архитектура сети представлена на рисунке 3:

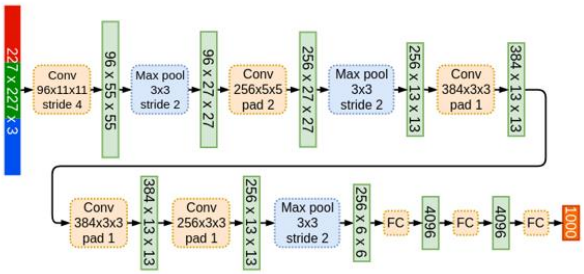


Рисунок 3. Архитектура сети AlexNet

Модель состоит из 8 слоёв: 5 свёрточных и 3 полносвязных. После каждого слоя, так же, как и в ResNet50, применяется функция активации ReLU. В модели используется LRN (Local Response Normalization), который помогает повысить обобщающую способность модели, имитируя биологическую активность нейронов. После некоторых свёрточных слоёв используется Max Pooling для уменьшения размерности, что помогает уменьшить переобучение. В полносвязных слоях используется Dropout для уменьшения переобучения. Этот метод заключается в том, что случайном образом, определённый процент нейронов сбрасывается, что делает сеть менее чувствительной к специфическим весам отдельных нейронов.

AlexNet является важным звеном в истории глубокого обучения. Успех этой сети показал большой потенциал свёрточных нейронных сетей в задачах компьютерного зрения.

В данной работе использовалась предобученная сеть AlexNet из torchvision.models. Последний слой был заменён для предсказания одного из 43 классов. Модель была дообучена. Предварительно были заморожены веса всех слоёв, кроме последнего. В качестве функции потерь использовался CrossEntropyLoss, а как оптимизатор был выбран Adam с learning rate равным 0.001. Процесс обучения состоял из 10 эпох.

V. Результаты

Модели обучались на части данных датасета GTSRB. Для обучения использовалось 2664 изображения дорожных знаков, а для теста – 12630 изображений.

Предсказания моделей оцениваются по следующим показателям:

- Precision: демонстрирует способность отличать классы дорожных знаков друг от друга.
- Recall: демонстрирует способность алгоритм обнаруживать классы дорожных знаков.
- F1: среднее гармоническое между precision и recall.

В результате предсказаний были получены следующие значения показателей:

ТАБЛИЦА 1. ОЦЕНКА МОДЕЛЕЙ

Показатель	***ResNet50***	***AlexNet***
Precision	0.6089594	0.5866203
Recall	0.5899446	0.5701504
F1	0.5755266	0.5632836

Результаты получились очень близкими друг к другу. ResNet50 оказался лучше AlexNet по каждому из показателей всего лишь на 1-2%. ResNet50 показал чуть более лучший результат, однако на само обучение AlexNet ушло гораздо меньше времени, что в данном контексте делает более практичным использование AlexNet.

VI. ЗАКЛЮЧЕНИЕ

В рамках данного исследования было проведено сравнение двух популярных архитектур нейронных сетей - ResNet50 и AlexNet - на задаче классификации дорожных знаков.

Эксперименты проводились на наборе данных GTSRB, содержащим изображения дорожных знаков Германии с различными вариациями. Для обучения использовалась небольшая часть данных, а для тестирования - значительно большая выборка.

Предобученные модели ResNet50 и AlexNet были адаптированы путем замены последнего слоя и дообучения на данных GTSRB. Обучение проводилось в течение 10 эпох с использованием оптимизатора Adam.

Для оценки качества моделей рассчитывались метрики precision, recall и F1. Результаты показали незначительное преимущество ResNet50 по всем метрикам (на 1-2%). Однако модель AlexNet обучалась гораздо быстрее.

Таким образом, с учетом небольшой разницы в метриках и различий во времени обучения, предпочтительнее использовать AlexNet для задачи классификации дорожных знаков. Хотя обе модели продемонстрировали сходную эффективность.

В целом, проведенное исследование позволило сравнить возможности двух популярных архитектур нейронных сетей для классификации изображений на примере задачи распознавания дорожных знаков. Полученные результаты могут быть полезны при выборе оптимальной модели для решения подобных прикладных задач.

ЛИТЕРАТУРА

[1] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[2] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[3] Ali, Bushra & Sadekov, Rinat. (2023). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation. 30. 87–105. 10.17285/0869-7035.00105.

[4] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[5] Smolin A, Yamaev A, Ingacheva A, Shevtsova T, Polevoy D, Chukalina M, Nikolaev D, Arlazarov V. Reprojection-Based Numerical Measure of Robustness for CT Reconstruction Neural Network Algorithms. Mathematics. 2022; 10(22):4210. https://doi.org/10.3390/math10224210 (Accessed: December 26, 2023).

[6] Efimoff, Albert & Matveev, Philip. (2022). Искусственный интеллект для науки и наука для искусственного интеллекта. Voprosy filosofii / Akademiia nauk SSSR, Institut filosofii. 95. 10.21146/0042-8744-2022-3-93-105.

[7] Mrinal Haloi, “Traffic Sign Classification Using Deep Inception Based Convolutional Networks”, arXiv, 2016, 1511.02992v2

[8] Amara Dinesh Kumar, “Novel Deep Learning Model for Traffic Sign Detection Using Capsule Networks”, arXiv, 2018, 1805.04424v1

[9] D. Temel, M-H. Chen, and G. AlRegib, ”Traffic Sign Detection under Challenging Conditions: A Deeper Look Into Performance Variations and Spectral Characteristics,” IEEE Transactions on Intelligent Transportation Systems, 2019.

[10] “GTSRB Dataset”, available at: https://benchmark.ini.rub.de/gtsrb_dataset.html (Accessed: December 26, 2023).

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, “Deep Residual Learning for Image Recognition”, arXiv, 2015, 1512.03385v

[12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2017. ImageNet classification with deep convolutional neural networks. Commun. ACM 60, 6 (June 2017), 84–90. https://doi.org/10.1145/3065386

[13] “PyTorch documentation. RESNET50”, available at: https://pytorch.org/vision/main/models/generated/torchvision.models.resnet50.html, (Accessed: December 26)

[14] “AlexNet. Wikipedia”, available at: https://en.wikipedia.org/wiki/AlexNet (Accessed: December 26, 2023).

Исследование возможности распознавания животных в искусственной среде

А. А. Ступина
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
stupinaaa99@gmail.com

***Аннотация* — с постоянным ростом городов и изменением природных сред, обеспечение устойчивости экосистем в городском ландшафте становится ключевым аспектом общественной и экологической ответственности. Распознавание животных в городской среде через применение передовых нейросетевых технологий имеет потенциал привнести положительные изменения в эту область. В данной работе было проведено исследование эффективности трансферного обучения для моделей, основанных на сверточных нейронных сетях – YOLOv8 и Faster R-CNN – в контексте задачи распознавания животных. В работе освещен процесс дообучения моделей на локальном наборе данных, собранном и аннотированном автором специально для обучения моделей распознаванию наиболее часто встречающихся в искусственных средах животных, а также проведено сравнение эффективности локализирующих и классифицирующей частей полученных моделей.**

Ключевые слова — компьютерное зрение, детекция животных, распознавание животных, YOLO, R-CNN mAP, COCO.

I. Введение

В последние годы наука значительно продвинулась в задаче распознавания самых разных объектов в связи с развитием глубокого обучения [1, 2, 3]. Крайне эффективным подходом оказалось использование трансферного обучения [4]. Детекторы, уже обученные на надежных и объемных наборах данных (в случае с YOLO [5] – на датасете COCO [6]) могут использоваться для распознавания объектов в более конкретных областях – например, для детектирования животных. Для этого модель «дообучается» на данных, специфичных для данной области, при этом обучение проходит только для нескольких последних слоев.

В этой работе будут рассмотрены технологии компьютерного зрения, которые могут помочь в решении задачи распознавания животных в условиях городской местности, а именно, будут исследованы возможность и эффективность трансферного обучения для таких моделей, как YOLOv8 [7] и Faster R-CNN [8]. Также эти модели достаточно быстрые, чтобы их можно было использовать и для детектирования объекта на потоке видео данных в реальном времени на машинах разумной мощности.

В качестве основных животных для обучения были выбраны собаки, кошки, еноты, олени и медведи. Они наиболее часто, среди прочих животных, забредают на территорию города, либо же обитают в его пределах, а также обладают достаточными размерами, чтобы их черты были различимы, например, на камерах уличного наблюдения. Отслеживание популяций кошек, собак, енотов может иметь большую значимость для общественного здоровья, т.к. они наиболее плотно контактируют с человеком, являясь при этом потенциальными разносчиками экто- и эндопаразитов, возбудителями кишечных заболеваний и бешенства. Это, безусловно, оказывает дополнительную нагрузку на городские санитарно-эпидемиологические службы. Обнаружение вторжения диких животных, например, медведей может позволить в краткие сроки принимать необходимые меры предосторожности и, при необходимости, привлекать внимание соответствующих служб. Также подобные системы детекции могут быть использованы как для поиска пропавших животных, и наблюдения за питомцами в пределах домашней территории. Такая технология может стать важной частью «умных городов» [9, 10].

Главной проблемой в решении данной задачи является нехватка существующих размеченных данных, соответствующих целевой области и необходимых для настройки предварительно обученных моделей. Несмотря на то, что многие общедоступные наборы данных (такие как COCO [6], ImageNet [11]) содержат изображения некоторых из животных, они не соответствуют всем необходимым требованиям, в частности городскому ландшафту, как среде, в рамках которой планируется детектировать объекты. В некоторых других наборах данных [12, 13, 14], специализированных на животных, представлены не все интересующие виды или отсутствуют ограничивающие рамки.

Существует исследование [15] детектирования животных в условиях городского ландшафта, однако оно больше сконцентрировано на обнаружении вторжения диких животных (в частности, специфичных для Канады), их набор данных для обучения содержит лишь одно из целевых животных – медведя и не соответствует целям данной работы.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались некоторые наборы данных, как локальные, собранные автором, так и открытые. Рассмотрим используемые открытые наборы.

A. COCO

COCO (Common Objects in Context) – это один из наиболее популярных и широко используемых датасетов в области компьютерного зрения. COCO был создан с целью содействия развитию алгоритмов для задачи обнаружения и сегментации объектов в изображениях. COCO включает в себя более 330 000 изображений, на которых размечены более 80 различных категорий объектов. Эти объекты охватывают разнообразные категории, такие как люди, животные, транспортные средства, еда и многие другие.

На рисунке 1 показаны примеры изображений COCO, взятые из меньшего набора данных в 1000 изображений, который был отобран исследователями из Стэндфордского университета. Он содержит всех представленных в COCO животных, а именно медведей, птиц, кошек, собак, жирафов, лошадей, овец и зебр. Для повышения эффективности обучения в датасете также преставлены проблемные случаи для детектирования животных, такие как:

- животное в движении, смазанное изображение (а);
- нестандартные ракурсы, в том числе слишком отдаленные (б)

Рис. 1. Примеры изображений целевых животных в наборе данных COCO

Рис. 2. Сложные случаи для детектирования: а) животное в движении б) нестандартный ракурс в) черно-белое изображение г) низкая освещенность

- монохромное изображение (в)
- изображения объектов при низкой освещенности (г)

B. ImageNet

Набор данных ImageNet [11] содержит 14 197 122 аннотированных изображения в соответствии с иерархией WordNet [16], количество изображений с аннотациями ограничительной рамки - 1 034 908. Набор данных ImageNet организован с использованием иерархии WordNet. Каждый узел в иерархии представляет категорию, и каждая категория описывается синсетом (набором синонимичных терминов). Изображения в ImageNet помечены одним или несколькими синсетами, предоставляя богатый ресурс для обучения моделей распознаванию различных объектов и их взаимосвязей. Всего в ImageNet используется 1000 синсетов, значительная часть которых представляет различные виды животных.

Набор данных ImageNet широко используется для обучения и оценки моделей глубокого обучения в различных задачах компьютерного зрения, таких как классификация изображений, обнаружение объектов и локализация объектов.

C. Локальный набор данных.

Для хранения изображений и аннотирования была выбрана платформа Roboflow [17]. Она позволяет вручную построить ограничивающие рамки на изображении и отнести их к тому или иному классу объектов, как показано на рисунке 3. В наборе данных присутствуют как сделанные авторами вручную снимки, так и найденные онлайн, при помощи Google Images и YouTube. Он содержит 1008 изображений целевых животных, на которых аннотировано: 255 медведей, 301 кот, 251 собака, 276 енотов и 254 оленя.

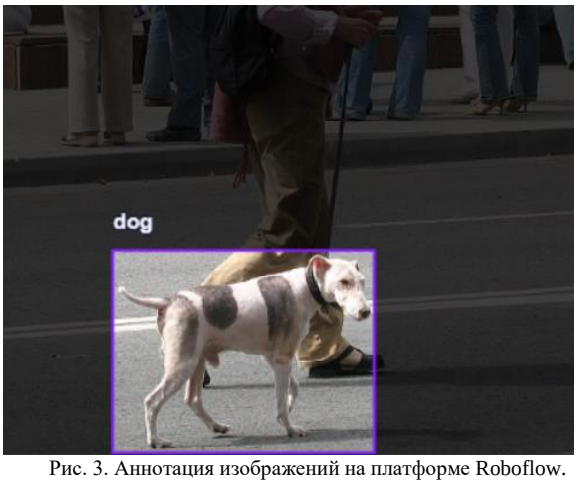


Рис. 3. Аннотация изображений на платформе Roboflow.

III. Нейросетевые архитектуры

A. YOLOv8

У YOLOv8 еще нет опубликованной статьи, поэтому пока имеется не так много данных о методах ее обучения и архитектуре. YOLOv8 использует сверточную нейронную сеть, которую можно разделить на две основные части: основную часть и «голову» [18]. Модифицированная версия архитектуры CSPDarknet53 [19, 20] составляет основу YOLOv8. Эта архитектура состоит из 53 сверточных слоев и использует частичные межэтапные соединения для улучшения потока информации между различными уровнями. «Голова» YOLOv8 состоит из нескольких сверточных слоев, за которыми следует ряд полностью связанных слоев. Эти слои отвечают за прогнозирование

ограничивающих рамок, оценки объектности и вероятности классов для объектов, обнаруженных на изображении.

YOLOv8 сначала разделяет входное изображение на сетку ячеек. Для каждой ячейки YOLOv8 прогнозирует набор ограничивающих рамок, а также вероятности классов для каждой ограничивающей рамки. Для решения проблемы перекрывающих друг друга рамок используется концепция IoU (Intersections over units). Основная ее цель – определить наиболее подходящую ограничивающую рамку. IoU измеряет перекрытие между прогнозируемой ограничивающей рамкой и истинной. Значение рассчитывается как отношение площади перекрытия между этими двумя рамками к общей площади, охватываемой их ими:

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union IoU}} \quad (1)$$

Значения IoU варьируются от 0 до 1. Значение 1 указывает на идеальное перекрытие между прогнозируемыми и фактическими ограничивающими рамками, а значение 0 означает отсутствие перекрытия между двумя рамками. В контексте обнаружения объектов более высокий IoU обычно означает более высокую точность локализации объектов на изображениях. Алгоритм игнорирует прогнозируемое значение ячейки сетки, имеющей низкое IoU. Однако, поскольку объект может быть связан с несколькими рамками, для которых значения IoU превышают установленный порог, далее применяется алгоритм подавления немаксимальных значений (NMS, non-maximum suppression).

Также YOLOv8 аугментирует изображения во время тренировочных циклов. Каждую эпоху модель видит несколько измененные варианты предоставленных ей изображений. Одна из таких аугментаций - мозаичная. Она включает в себя склеивание четырех изображений вместе. Это повышает эффективность детектирования моделью объектов в новых местах, в частичной окклюзии и на фоне разных окружающих пикселей. Было показано [21] что мозаичная аугментация снижает эффективность модели, если применяется в течении всего процесса обучения, поэтому она применялось лишь в ходе последних 10 эпох обучения. Результатом работы модели является определение местоположения и класса (одного из 80) объектов на изображении.

COCO — это отраслевой стандарт оценки моделей обнаружения объектов. При сравнении моделей на COCO в качестве функции потерь обычно используется mAP (mean Average Precision), рассчитанная как для всей выборки в целом, так и для каждого класса. Она считается как площадь под кривой precision-recall.

$$\text{mAP} = \frac{1}{\text{N}}\sum_{\text{i}=1}^{\text{N}} \text{AP}_\text{i} \quad (2)$$

YOLOv8 достигает высокой точности на COCO. Например, модель YOLOv8m — средняя модель — достигает 50,2% mAP при измерении на COCO. Всего существует 5 вариантов модели YOLOv8 (таблица 1).

Для данной работы дообучение решено было проводить для модели YOLOv8m, как оптимальной с точки зрения баланса скорости и точности. В обучаемой модели 295 слоев. При этом 469 из 475 весов переносятся из ранее обученной модели, таким образом в процессе дообучения изменяются только 6 из них.

ТАБЛИЦА I. Вариации модели YOLOv8

Модель	Размер (пиксели)	mAP^{val} (50-95)	Скорость A100 TensorRT (мс)	Параметры (M)
YOLOv8n	640	37.3	0.99	8.7
YOLOv8s	640	44.9	1.2	28.6
YOLOv8m	640	50.2	1.83	78.9
YOLOv8l	640	52.9	2.39	165.2
YOLOv8x	640	53.9	3.53	257.8

B. Faster R-CNN

Архитектура Faster R-CNN (Region-based Convolutional Neural Network) представляет собой метод для обнаружения объектов в изображениях [8]. Он был представлен Россом Гиршиком на конференции Computer Vision and Pattern Recognition (CVPR) в 2015 году. Faster R-CNN внесла существенный вклад в область обнаружения объектов, показывая высокую точность и эффективность работы.

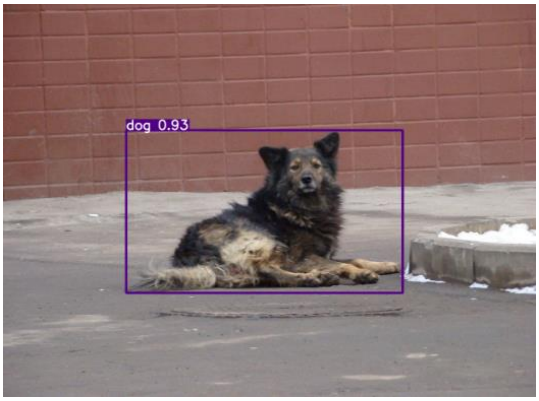


Рис. 5. Пример работы модели: при помощи алгоритмов IoU и NMS определяется наиболее подходящая для объекта рамка, а также указывается «уверенность» модели в классе объекта.

	Type	Filters	Size	Output
	Convolutional	32	3 × 3	256 × 256
	Convolutional	64	3 × 3 / 2	128 × 128
1×	Convolutional	32	1 × 1	
	Convolutional	64	3 × 3	
	Residual			128 × 128
	Convolutional	128	3 × 3 / 2	64 × 64
2×	Convolutional	64	1 × 1	
	Convolutional	128	3 × 3	
	Residual			64 × 64
	Convolutional	256	3 × 3 / 2	32 × 32
8×	Convolutional	128	1 × 1	
	Convolutional	256	3 × 3	
	Residual			32 × 32
	Convolutional	512	3 × 3 / 2	16 × 16
8×	Convolutional	256	1 × 1	
	Convolutional	512	3 × 3	
	Residual			16 × 16
	Convolutional	1024	3 × 3 / 2	8 × 8
4×	Convolutional	512	1 × 1	
	Convolutional	1024	3 × 3	
	Residual			8 × 8
	Avgpool		Global	
	Connected		1000	
	Softmax			

Рис. 6. Архитектура Darknet-53, лежащая в основе CSPDarknet53

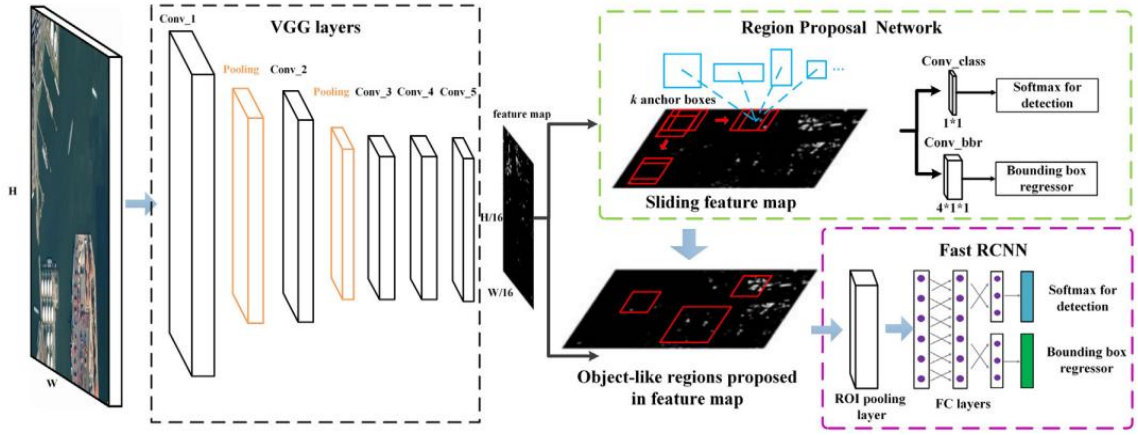


Рис. 7. Архитектура Faster R-CNN на основе VGG

Рассмотрим основные компоненты архитектуры Faster R-CNN (рисунок 7).

- Convolutional Backbone (Backbone сеть). Faster R-CNN использует предварительно обученную сверточную нейронную сеть, такую как VGG16 [22], ResNet [23], или другие, в качестве основы (backbone) для извлечения признаков из входного изображения [24]. Эта сеть служит для выделения различных уровней абстракции. VGG16 обучена на подмножестве набора данных ImageNet.
- Region Proposal Network (RPN). RPN является ключевой частью Faster R-CNN [8]. Он используется для генерации возможных областей (region proposals), в которых могут находиться объекты. RPN работает на выходе сверточных слоев backbone сети и предлагает прямоугольные области, которые затем используются для дальнейшего обнаружения объектов.
- RoI (Region of Interest) Pooling – после получения координат областей от RPN, используется RoI Pooling для выравнивания областей разного размера в фиксированный размер [24]. Это необходимо для того, чтобы передать эти области как вход в последующие слои нейронной сети.

После RoI Pooling обработанные области подаются на полностью связанный слой (fully connected layer), который в конечном итоге предсказывает класс объекта и его прямоугольные координаты внутри RoI. Также, как и в YOLOv8, применяется алгоритм подавления немаксимальных значений для уменьшения количества дубликатов и отсева менее точных предсказаний [18]. Он удаляет избыточные предсказания, оставляя только наиболее уверенные и неперекрывающиеся области.

Преимущества Faster R-CNN включают эффективное использование общей архитектуры для генерации кандидатов областей и дальнейшего обнаружения объектов, что позволяет достигнуть высокой точности при относительно низкой вычислительной стоимости по сравнению с предыдущими методами.

IV. Сравнение

Было проведено обучение моделей YOLOv8 и Faster R-CNN. Для обучения использовался локальный набор данных, который был разбит на тренировочную, тестирующую и валидационную выборки в соотношении 7:1:2. Качество работы модели оценивалось как для локализации объекта, так и для его классификации. Использовались следующие меры:

- TP – модель обнаружила животное там, где оно действительно есть.
- FP – модель обнаружила животное там, где его нет.
- FN – модель не обнаружила животное, хотя оно присутствует на изображении.

По введённым величинам строятся такие функции оценок, как:

- Точность – сколько раз модель обнаружила животное там, где оно действительно есть к общему числу детектированных животных:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \quad (3)$$

- Полнота – сколько животных обнаружила модель от общего числа животных:

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (4)$$

- F1-мера – гармоническое среднее между точностью и полнотой, если один из параметров стремиться к нулю, она также стремиться к нулю:

$$\text{F1} = \frac{2\times\text{Precision}\times\text{Recall}}{\text{Precision+Recall}} \quad (5)$$

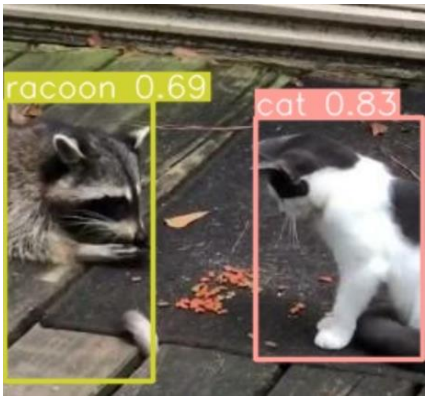


Рис.8. Пример работы обученной модели YOLOv8

Так же использовалась mAP метрика. Она учитывает соотношение между точностью и полнотой, беря во внимание как ложноположительные (FP), так и ложноотрицательные результаты (FN). В соответствии с

рекомендациями COCO 2017-го года по оценке задач, mAP рассчитывался путем усреднения AP по всем классам объектов и по всем 10 пороговым значениям IoU от 0,5 до 0,95 с размером шага 0,05.

В таблице 3 показаны количественные характеристики двух используемых подходов [25, 26]. Оба подхода имеют достаточно высокие показатели для исследуемых метрик, однако Faster R-CNN показывает себя лучшим детектором из этих двух, имея лучшие показатели точности и обнаруживая большее количество животных.

ТАБЛИЦА III. Сравнение детектирующей части.

	YOLOv8	Faster R-CNN
Precision	0.898	0.894
Recall	0.785	0.915
F1	0.838	0.904
mAP50	0.863	0.908
mAP50-95	0.68	0.676

Результаты оценки классифицирующей части можно видеть на рисунках 9 и 10. По представленным матрицам ошибок можно сказать, что для обоих классификаторов наиболее сложным объектом для детектирования являются кошки, но в целом Faster R-CNN показывает меньше ошибок при классификации объектов.

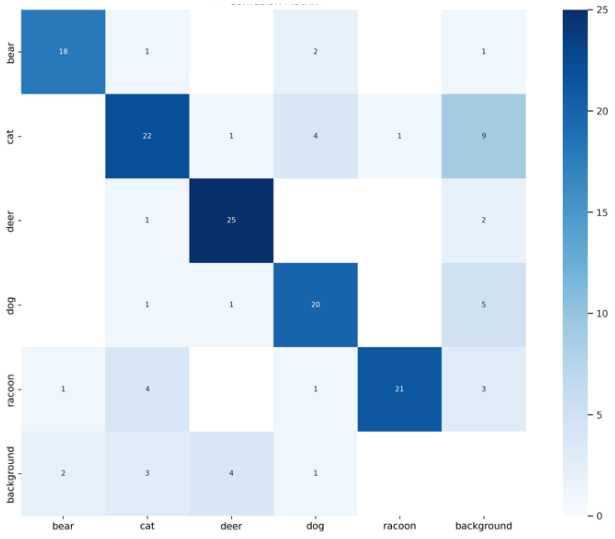


Рис.9. Матрица ошибок работы YOLOv8.

Лучшие показатели качества Faster R-CNN можно объеснить тем, что данная модель включает в себя большее число параметров, кроме того она обучалась на датасете ImageNet, в котором более широко представлены различные виды животных, чем в наборе данных COCO. Также само дообучение модели Faster R-CNN проходит дольше в сравнении с YOLOv8.

Однако, несмотря на то, что Faster R-CNN может обеспечить более высокую точность обнаружения, особенно на небольших объектах и в случае наличия перекрывающихся объектов, этот подход как правило более медленный, чем YOLO [27].

V. Заключение

В данной работе были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети – COCO и ImageNet. Была проведена оценка двух наиболее эффективных подходов к детектированию – YOLOv8 и Faster R-CNN, а также рассмотрена их архитектура.

Данные модели были дообучены на собранном и размеченном автором наборе данных, после чего, проведено сравнение их эффективности детектирования и классификации.

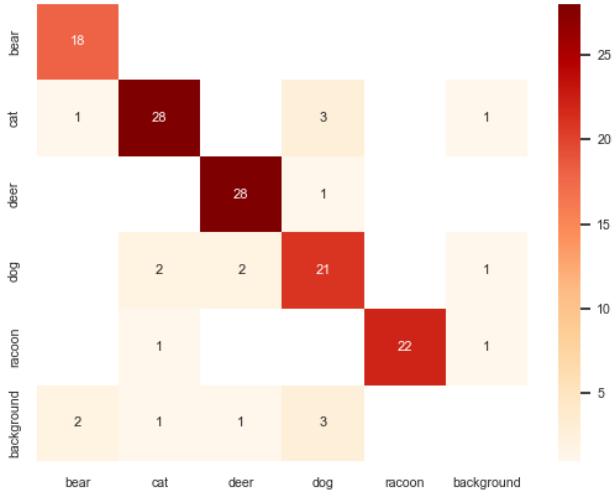


Рис.10. Матрица ошибок работы Faster R-CNN.

По полученным оценкам очевидно, что подход Faster R-CNN превосходит YOLOv8 по точности детектирования и классификации. Однако данная модель требует большего количество времени как для обучения, так и для работы. В связи с этим можно предположить, что для использования детекторов в режиме реального времени модель YOLOv8 может быть предпочтительнее, так как незначительное снижение показателей относительно Faster R-CNN она компенсирует более высокой скоростью работы.

Литература

[1] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[2] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[3] Arlazarov, V.L., Arlazarov, V.V., Bulatov, K.B. et al. Mobile ID Document Recognition–Coarse-to-Fine Approach. Pattern Recognit. Image Anal. 32, 89–108 (2022). https://doi.org/10.1134/S1054661822010023

[4] S.J. Pan and Q. Yang. “A survey on transfer learning”. IEEE Transactions on Knowledge and Data Engineering, 22:1345–1359, 2010.

[5] J. Redmon, S. Divvala, R. Girshick, A. Farhadi. “You Only Look Once: Unified, Real-Time Object Detection”, Facebook AI Research, 2016

[6] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. “Microsoft coco: Common objects in context”. In D. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, editors, ECCV 2014, pages 740–755, Cham, 2014. Springer International Publishing

[7] “YOLOv8: Pioneering Breakthrought in Object Detection Technology” available at: https://blog.paperspace.com/yolov8-a-

revolutionary-advancement-in-object-detection-2/ (Accesed: December 4, 2023)

[8] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1, NIPS'15, pages 91–99, Cambridge, MA, USA, 2015. MIT

[9] S. V. Solodov et al. Framing regional innovation and technology policies for transformative change. 2022. IOP Conf. Ser.: Earth Environ. Sci. 981 022007. DOI 10.1088/1755-1315/981/2/022007

[10] Y. S. Chernyshova, B. I. Savelyev, S. V. Solodov et al. Applying distributed ledger technologies in megacities to face anthropogenic burden challenges. 2022. IOP Conf. Ser.: Earth Environ. Sci. 1069 012028. DOI 10.1088/1755-1315/1069/1/012028

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. "ImageNet: A Large-Scale Hierarchical Image Database". In CVPR, 2009.

[12] Labeled information library of alexandria: Biology and conservation. online: http://lila.science/datasets1, August 2019.

[13] A. Swanson, M. Kosmala, C. Lintott, R. Simpson, A. Smith, and C. Packer. "Snapshot serengeti, highfrequency annotated camera trap images of 40 mammalian species in an african savanna". Scientific Data, 2:150026–, June 2015.

[14] J. Parham, C. Stewart, J. Crall, D. Rubenstein, J. Holmberg, and T. Berger-Wolf. "An animal detection pipeline for identification". In WACV, 2018.

[15] A.Singh, M.Pietrasik, G. Natha et al. "Animal Detection in Man-made Environments", IEEE Xplore, 2020 pp.1438-1449

[16] Christiane Fellbaum (1998, ed.) WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.

[17] Dwyer, B., Nelson, J. (2022), Solawetz, J., et. al. Roboflow (Version 1.0) [Software]. Available from https://roboflow.com. computer vision.

[18] D. Reis, J. Kupec, J. Hong et al. "Real-Time Flying Object Detection with YOLOv8", Georgia Institute of Technology, 2023

[19] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," CoRR, vol. abs/1804.0, 201

[20] K. Simonyan, A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition", The 3rd International Conference on Learning Representations (ICLR2015), 2014, pp. 1-14.

[21] Z. Wei, C. Duan, X. Song, Y. Tian, H. Wang "Amrnet: chip augmentation in aerial image object detection", Shchool of Computer Science and Technology, 2020

[22] K. He, X. Zhang, S. Ren, J. Sun. "Deep Residual Learning for Image Recognition", Microsoft Research, 2015

[23] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[25] [26] Z. C. Lipton, C. P.Elkan, B. Narayanaswamy. "Thresholding Classifiers to Maximize F1 Score", 2014 arXiv: Machine Learning, pp. 1-16.

[26] M. Sokolova, N. Japkowicz, S. Szpakowicz. "Beyond accuracy, F-score and ROC: a family of discriminant measures for performance evaluation", Proceedings of Australasian joint conferenceon artificial intelligence, 2006, vol. 4304, pp. 1015-1021.

[27] J. -a. Kim, J. -Y. Sung and S. -h. Park, "Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition," 2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia), Seoul, Korea (South), 2020, pp. 1-4, doi: 10.1109/ICCE-Asia49877.2020.9277040.

Анализ подходов к использованию предобученных моделей в разработке корпоративных чат-ботов

Д.В. Береснев
Кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2314671@edu.misis.ru

Аннотация — в данной работе рассматривается использование предобученных моделей с архитектурой трансформер при разработке корпоративных чат-ботов, основанных на базе знаний. Были рассмотрены BERT и GPT модели, а также их применение в решении различных NLP-задач. С помощью наборов данных в исследовании сымитированы запросы пользователей и база знаний компании. Реализован семантический поиск по документам на основе алгоритма k-ближайших соседей и различных предобученных моделей, базирующихся на BERT. Проведено сравнение. Для решения задачи классификации на предварительно подготовленном небольшом наборе данных проведены испытания, для определения наилучшей архитектуры модели и оптимальных параметров обучения. Проведены тестовые испытания с полным набором данных и увеличенным количеством эпох. На основании результатов определены наиболее оптимальные настройки модели для решения задачи классификации. Рассмотренные подходы решения задач семантического поиска и классификации легли в основу предложенной архитектуры чат-бота. Также были представлены варианты масштабирования системы.

Ключевые слова — обработка естественного языка, предобученные модели, трансформеры, семантический поиск, классификация текста, BERT, GPT, hugging face, sentence transformers, чат-бот, база знаний, тонкая настройка

I. Введение

Нейронные сети — это мощный инструмент искусственного интеллекта, который развивает множество отраслей благодаря своей способности извлекать сложные закономерности из огромного объема данных. В прогнозировании поведения транспортных средств они помогают повышать безопасность и эффективность движения, в здравоохранении нейронные сети способствуют диагностике, в робототехнике они стоят за развитием автономных систем, а в томографии играют ключевую роль в улучшении качества и скорости обработки изображений [1], [2], [3], [4]. Приведенные способы применения подтверждают актуальность нейронных сетей в современном мире. Переходя на область обработки естественного языка и баз знаний, данные технологии могут трансформировать способы, с помощью которых компании поддерживают, развивают и используют необходимую им информацию.

Большинство компаний выделяют средства на создание и поддержку корпоративной базы знаний, что, в теории, должно снизить количество ошибок, привести рабочий процесс к общим стандартам, упростить интеграцию новых сотрудников и рациональнее использовать время команды [5], [6], однако, часто встречается проблема неэффективного использования базы знаний в компаниях [7], [8]. Несмотря на наличие информации, сотрудники не всегда могут или желают использовать ее, что снижает общую работоспособность системы.

Решением данной проблемы может быть использование языковых моделей, обученных на корпоративной базе знаний [9], [10], однако, разработка подобных моделей с нуля может быть неразрешимой задачей даже для крупных IT-компаний, поскольку это требует огромного количества времени и вычислительных ресурсов, а также необходимости подключения высококвалифицированных специалистов [11]. Чтобы избежать данных проблем и целесообразно использовать ресурсы компании, предлагается базировать разработку на предобученных моделях, которые изначально справляются с большинством типов задач обработки естественного языка [12], и дообучить их на корпоративных данных с целью создания микросервисов, облегчающих доступ к искомой информации и предоставляющих краткий и информативный ответ.

Требования, которые должны быть покрыты разработанной системой, могут включать решение следующих типов задач:

- Семантический поиск. Позволяет отобразить список документов, которые с наибольшей вероятностью содержат ответ на вопрос.
- Классификация текста. Может быть использована для определения отдела или группы ответственных лиц, причастных к теме заданного вопроса.
- Ответ на вопрос. Поиск предложения в тексте документа, отвечающего на заданный вопрос
- Краткое изложение текста. Генерация новой текстовой последовательности, позволяющая дать краткое изложение найденного документа.
- Генерация текста. Формирование связанного ответа на основе полученной информации.

Для решения каждого типа задачи необходимо выбрать наиболее подходящую предобученную модель и с помощью тонкой настройки дообучить на корпоративном наборе данных с применением методов, наиболее оптимальных для выбранного типа. Набор обученных моделей может быть объединен в единую систему, которая выполняет поставленные требования. Попытка обучения одной универсальной модели, покрывающей все типы задач, является нецелесообразной, а также показывает в среднем худшие результаты [13].

II. Наборы данных

A. SQuAD

The Stanford Question Answering Dataset — один из самых популярных и широко используемых наборов данных в области обработки естественного языка. Представляет собой набор вопросов и ответов, связанных с отрывками статей из Википедии. Каждый пример в наборе данных SQuAD состоит из отрывка текста и вопроса, связанного с этим текстом.

Данный набор данных хорошо подходит для имитации семантического поиска по документам, где в роли документов выступает список уникальных текстов, содержащих ответ, а в роли запросов — список вопросов датасета.

B. Stackoverflow Question Classification Challenge

Общедоступный набор данных, состоящий из более чем 80 тысяч заголовков вопросов, взятых непосредственно из StackOverflow с использованием их API. К каждому вопросу прилагается список тегов и язык программирования, о котором задан вопрос. Таким образом, датасет представляет набор, разделенный на 5 классов:

- Python
- R
- Java
- JavaScript
- PHP

C. Персональный набор данных

Персональный датасет включает в себя 200 записей, содержащих контекст (имитация статьи или инструкции в базе знаний), запрос, на который отвечает связанный с ним контекст (имитация вопросов пользователя) и лейбл, который категоризирует запрос и контекст по одному из классов Stackoverflow Question Classification Challenge. Набор данных был собран путем парсинга сайтов документации по языкам программирования, генераций модели GPT-4 и ручного заполнения. Данный датасет использовался для тестирования алгоритма семантического поиска и обученной модели классификации.

III. Предобученные модели

В качестве предобученного трансформера для решения задачи классификации был выбран DistilBERT. Это небольшая, быстрая, и нетребовательная модель, обученная на базе BERT. В сравнении с BERT, DistilBERT имеет на 40% меньше параметров, работает на 60 % быстрее, сохраняя при этом более 95% производительности BERT, измеренной в тесте понимания языка GLUE [14].

IV. Среда выполнения

В ходе исследования все необходимые вычисления, анализ данных и разработка алгоритмов были выполнены в интерактивной среде Google Colab, которая предоставляется Google Research. Google Colab позволяет писать и выполнять код Python в браузере с доступом к вычислительным ресурсам, включая графические процессоры (GPU). На Рисунке 1 представлены основные параметры GPU, который был использован в процессе исследований.

```
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 535.104.05             Driver Version: 535.104.05   CUDA Version: 12.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |         Memory-Usage | GPU-Util  Compute M. |
|                                         |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla T4                       Off | 00000000:00:04.0 Off |                    0 |
| N/A   58C    P8              10W /  70W |      0MiB / 15360MiB |      0%      Default |
|                                         |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
```

Рис. 1. Характеристики GPU среды Google Colab

V. Классический подход к обработке естественного языка

Обработка естественного языка (NLP) является одной из самых сложных задач в области глубокого обучения, поскольку она связана с созданием моделей, которым необходимо работать со множествами языков, определять неоднозначность и контекстуальную зависимость в текстах, обрабатывать длинные последовательности и генерировать новый текст. В области обработки естественного языка также определен широкий спектр разнообразных задач, которые должны быть покрыты такими моделями. Подобное множество проблем влияет на сложность моделей по данному направлению.

Классический подход в области NLP включает следующие этапы [15], [16]:

1. Предобработка текста. Текст чистится от неинформативных символов, проводится токенизация (разбиение текста на отдельные слова или токены), удаляются стоп-слова, применяется стемминг или лемматизация для приведения слов к базовой форме.
2. Векторизация. Слова или токены преобразуются в числовые векторы с помощью различных методик. Наборы полученных векторов определяют признаки входных предложений и подаются на вход нейронной сети
3. Построение модели. Создается архитектура нейронной сети, которая может включать в себя сверточные и рекуррентные нейронные сети или использовать более современный подход - архитектуру трансформеров
4. Обучение модели. Нейронная сеть обучается на наборах текстовых данных. С помощью размеченных классов или маскирования определенных слов формируются веса, которые позволяют определять семантику входных предложений и решать задачи обработки языка
5. Оценка и тонкая настройка. на основании тестовой выборки принимается решения о внесении изменений в архитектуру или гиперпараметры для улучшения производительности модели

Достижение высоких результатов с использованием классических подходов к построению моделей является невозможным или нецелесообразный для большинства компаний. Альтернативной в данном случае является использование предобученных моделей.

VI. Предобученные модели

Предобученные модели заменяют первые четыре пункта построения нейронной сети для решения задач естественного языка (токенизация, векторизация, построение и обучение модели), предоставляя подготовленную базу для тонкой настройки под конкретные нужды [17]. Сложная архитектура и предварительное

обучение на огромных наборах данных делают их гибким и универсальным инструментом.

BERT (пункт A) и GPT (пункт B) представляют собой предобученные модели, базирующиеся на технологии трансформеров.

Трансформеры — это класс моделей глубокого обучения, использующий механизм внимания [18]. Внимание — сегмент, который позволяет динамически фокусироваться на различных частях входных данных, определяя важность каждого слова исходя из контекста [18], [19]. В совокупности с другими слоями архитектуры данный подход позволяет содержать информацию о всей последовательности для каждого слова [18], [19]. Это, например, наделяет абстрактные слова «это» или «он» семантикой данного предложения (Рисунок 2).

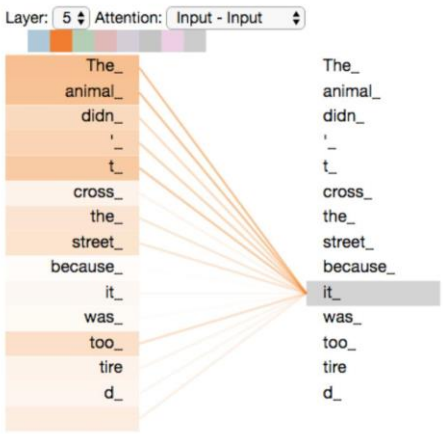


Рис. 2. Демонстрация работы механизма внимания

Трансформеры представляют собой последовательное размещение стека эндкодеров и декодеров, преобразующих одну последовательность в другую (Рисунок 3) [20].

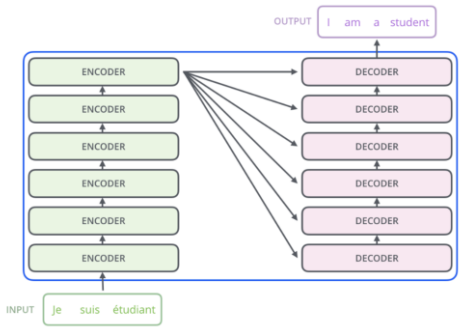


Рис. 3. Структура архитектуры трансформера

Каждый элемент включает в себя слой внимания и прямого распространения (Рисунок 4) [20].

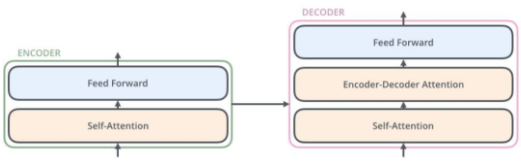


Рис. 4. Структура энкодеров и декодеров

A. *BERT-модели*

BERT (Bidirectional Encoder Representations from Transformers) представляет собой набор энкодеров, который переводит входную последовательность в набор эмбеддингов (векторное представление), при этом, каждое слово последовательности будет нести информацию о всей последовательности (такое поведение частично описано фразой «Bidirectional Encoder» в названии). Таким образом, формирование эмбеддингов зависит от контекста, что позволяет семантически разделять одинаковые по написанию слова и формировать точные векторные представления для слов, которые без контекста не имеют особого значения [20].

Поскольку выходом модели BERT является скрытый слой векторного представления входной последовательности, а обучался BERT на задачах маскирования и классификации двух предложений, он идеально подходит для тонкой настройки под специфичные задачи, в особенности различных типов классификации, ответов на вопросы и может быть использован для семантического поиска, что покрывает большую часть поставленных задач для разработки корпоративного чат-бота.

B. *GPT-модели*

GPT-модели используют обратный принцип и представляют собой стек декодеров [21]. Подобный подход отлично подходит для генерации текстов и предсказания следующего элемента последовательности, однако, используя некоторые трансформации в архитектуре модели, можно добиться решения задач классификации, логического следствия, схожести и множественного выбора, но результаты в подобных типах задач будут уступать BERT-моделям [22]. Также GPT-модели воспринимают последовательность слева направо, в отличии от двусторонней связи в BERT, что влияет на менее плотные семантические связи в эмбеддингах.

Результируя вышесказанное, можно заключить, что BERT-модели лучше подходят для задач, где важно понимание текста, а GPT-модели отлично справляются с задачами генерации и предсказания. В целом, оба типа моделей могут быть полезны для покрытия поставленных задач разработки корпоративного чат-бота, но для высокой результативности стоит в первую очередь обратить внимание на модели по типу BERT.

Для поиска и использования предобученных моделей можно воспользоваться платформой Hugging Face, которая предоставляет различные вариации трансформеров, обученных на всевозможных наборах данных под разные задачи. Платформа также предоставляет датасеты, инструменты для тонкой настройки и документацию. С помощью Hugging Face можно быстро опробовать и установить «сырые» модели и модели, которые «из коробки» хорошо справляются с большим спектром NLP-задач на разных языках.

VII. Семантический поиск

Семантический поиск — это процесс поиска информации, который ориентирован на выявление смысловой близости между запросом пользователя и ответом, а не на поиск точных совпадений между словами или фразами [23].

Традиционные решения поиска в тексте, такие как коэффициент Жаккара, алгоритм шинглов и расстояние Левенштейна, определяют некоторое число на основе слов или фрагментов текста, которое отображает схожесть запроса и ответа, однако мера схожести не отображает семантическую близость и часто будет выдавать ложные ответы из-за разных формулировок, контекста,

вводных и связывающих слов и фраз, а также распространенных выражений [24]. Частично, данную проблему решает TF-IDF поиск и его более современные разновидности. Данный подход основывается не только на количестве и частоте слов запроса в искомом тексте, но также учитывает уникальность этих слов [25]. Таким образом, для каждого слова из запроса формируется некоторая «важность» на основе частоты упоминаний в документе, которая затем будет влиять на результаты поиска. Подобный подход справляется с полнотекстовым поиском значительно лучше, но не учитывает семантику запроса, оставляя следующие проблемы:

- Поиск не группирует в одну категорию слова, одинаковые по значению, но разные по написанию
- Поиск группируют в одну категорию слова одинаковые по написанию, но разные по значению
- Поиск не учитывает контекстную значимость, которая может определять важность слов в запросе

Использование предобученных моделей BERT решает эти проблемы [26], [27]. Поскольку выходом сети является набор эмбеддингов, представляющих сжатое семантическое представление входной текстовой последовательности в виде векторов в пространстве, мы можем находить близкие вектора, и, соответственно, похожие по смыслу конструкции.

Набор данных SQuAD имитировал базу знаний и вероятные запросы пользователей. Предварительно были исключены все повторяющиеся контексты. Получив около девятнадцати тысяч уникальных пар вопрос-контекст, необходимо было проанализировать возможные длины текстов (Рисунок 5), поскольку BERT-модели обычно принимают не более пятисот двенадцати слов. В данном наборе проблем с длинной текстов не было, в ином же случае, можно было прибегнуть, например, к разделению длинных документов.

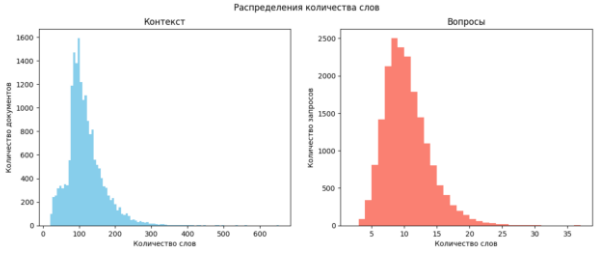


Рис. 5. Длины текстов датасета SQuAD

Для оценки способности модели к семантическому поиску с помощью алгоритма k-ближайших соседей для каждого из девятнадцати тысяч вопросов формировались группы из одного, трех, пяти и десяти документов, которые были ближе всего к запросу в пространстве (пример для группы из одного документа представлен на Рисунке 6) [28].

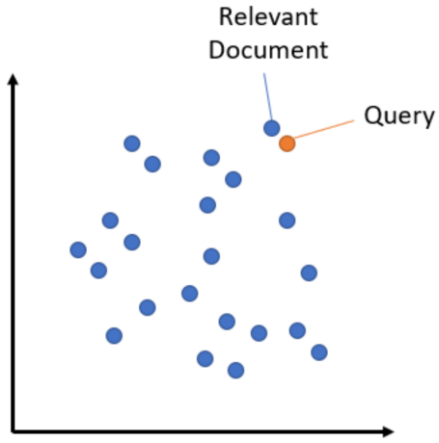


Рис. 6. Поиск ближайшего к запросу документа

Для измерения близости использовалось скалярное произведение (формула 1) и косинусоидальное сходство (формула 2) [29].

$$(1) A \cdot B = \sum(A_i B_i)$$

$$(2) Sim(A,B) = cos\theta = \frac{A \cdot B}{||A||||B||}$$

Вышеперечисленные действия были выполнены для каждого запроса. Поскольку набор данных однозначно связывает один документ и один запрос в наборе, после 18877 итераций были получены данные, отображающие процент попадания документа, содержащего ответ на вопрос в каждую группу из одного, трех, пяти и десяти документов. Перевод текстовой последовательности в векторное пространство был осуществлен с помощью токенизатора DestilBERT, модели DestilBERT, а также нескольких моделей из библиотеки Sentence Transformers.

Первым делом был осуществлен перевод текста в пространство с помощью одного токенизатора. Результаты представлены в Таблице 1.

Таблица 1 — Результаты семантической точности для токенизатора DestilBERT

	Топ 1	**Топ 3**	**Топ 5**	**Топ 10**
Точность (%)	0.0023	0.0031	0.0032	0.0052

Настолько низкая точность обусловлена тем, что токенизатор не формирует семантические отношения между словами в предложениях. Близость векторов в пространстве формируется исключительно из набора одинаковых слов.

После этого корпус текстов и вопросов был переведен в векторное пространство с помощью предобученной модели DestilBERT. Результаты представлены в Таблице 2.

Таблица 2 — Результаты семантической точности для DestilBERT

	Топ 1	**Топ 3**	**Топ 5**	**Топ 10**
Точность (%)	0.52	0.81	0.98	1.23

Низкая точность является следствием использования «сырой» версии BERT без тонкой настройки, однако существует ряд других проблем. Во-первых, созданные трансформером эмбеддинги для текстовой последова-

тельности представляют собой сотни векторов, что значительно усложняет процесс сравнения. Во-вторых, для обработки последовательностей, все данные должны быть переданы в сеть, что ведет к значительным затратам вычислительных ресурсов и ресурсов памяти. Если речь идет о поиске похожих пар в коллекции из десяти тысяч предложений, то модели типа BERT может понадобиться порядка шестидесяти пяти часов на выполнение задачи [30]. Поэтому, для задач семантического поиска рекомендуется использовать трансформеры типа Sentence-BERT. Это модификация модели BERT, специально обученная для кодирования целых предложений в вектор фиксированного размера. Это достигается за счет использования siamese и triplet network структур, которые позволяют эффективно сравнивать предложения. Для сравнения, Sentence-BERT модель справится с задачей поиска пар из десяти тысяч предложений не за шестьдесят пять часов, а за пять секунд, затрачивая меньше ресурсов и сохраняя при этом точность BERT [30].

Таким образом, для оптимального решения задачи семантического поиска модель DestilBERT была заменена на модели из библиотеки Sentence Transformers. Поскольку данная задача относится к категории асинхронного поиска (нахождение длинного абзаца по короткому запросу), были выбраны только те модели, которые изначально обучались для асинхронного поиска. Также среди оставшегося множества предоставляемых библиотекой предобученных моделей были выбраны те, которые базировались на DistilBERT и были обучены с использованием косинусоидального сходства или скалярного произведения (эти данные учитывались при выборе способа измерения близости в алгоритме k-ближайших соседей).

После измерений были проведены сравнения. На Рисунке 7 представлены результаты поиска по тестовой базе знаний для моделей msmarco-distilbert-base-v3 (использует косинусоидальное сходство), msmarco-distilbert-base-dot-product-v3 (использует скалярное произведение), msmarco-distilbert-base-tas-b (использует скалярное произведение). Для каждой предобученной модели было сделано 18877 итераций работы алгоритма k-ближайших соседей. Собранные данные демонстрируют процент содержания документа с ответом на поставленный вопрос в предсказываемой группе из одной, трех, пяти и десяти статей среди всех документов базы знаний.

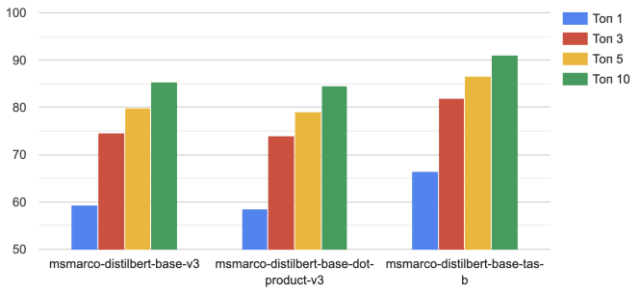


Рис. 7. Анализ работы DistilBERT-моделей Sentence Transformers

Также были проведены замеры предобученных моделей из библиотеки Sentence Transformer, базирующиеся на отличных от DistilBERT трансформерах. Итоговая таблица всех измерений точности семантического поиска по 18877 документам для групп в один, три, пять и десять документов представлена в Таблице 3.

Таблица 3 — Итоговые результаты точности инструментов векторизаци текста для задачи семантического поиска.

	Попадание в группу из 1 документа	**Попадание в группу из 3 документов**	**Попадание в группу из 5 документов**	**Попадание в группу из 10 документов**
Токенизатор	0.0023%	0.0031%	0.0032%	0.0052%
DestilBERT	0.52%	0.81%	0.98%	1.23%
Msmarco-distilbert-base-v3	59.38%	74.55%	79.78%	85.37%
Msmarco-distilbert-base-dot-prod-v3	58.54%	73.98%	79.08%	84.62%
Msmarco-distilbert-base-tas-b	66.54%	81.97%	86.53%	91.10%
all-MiniLM	58.47%	76.89%	82.60%	88.99%
Msmarco-distilroberta-base-v2	59.38%	74.55%	79.78%	85.37%
Msmarco-distilbert-base-v2	56.86%	73.72%	79.20%	85.24%
multi-qa-MiniLM-L6-cos-v1	60.73%	77.64%	83.22%	88.81%
Msmarco-roberta-base-ance-firstp	54.90%	69.96%	75.09%	80.60%
Msmarco-distilbert-base-v4	52.38%	69.95%	75.91%	82.78%

Таким образом, основываясь на запросе, модели могут показать пользователю пять статей из девятнадцати тысяч и, в среднем, с восьмидесятью процентной вероятностью в предоставленном наборе будет содержаться ответ на поставленный вопрос. Уменьшение общего количества документов значительно повлияет на увеличение процента содержания ответа. Это является отличным показателем, учитывая, что была использована готовая модель, которая обучалась на других наборах данных. Среди всех испытанных предобученных моделей лучше всего себя показала Msmarco-distilbert-base-tas-b.

Результаты также можно улучшить дообучением модели на персональном наборе данных. Для этого, в процессе обучения, можно проверять предсказания модели и, в случае отсутствия верных документов, понижать необходимые веса. Повторяя этот процесс, модель будет лучше справляться с поиском. Более подробно процесс дообучения трансформеров описан в главе о классификации текста.

VIII. Классификация текста

Для классификации текста был использован набор данных Stackoverflow Question Classification Challenge. Структура датасета позволяет имитировать сценарий, в

котором модель классифицирует запрос среди небольшого количества классов (Рисунок 8).

	title	id_stack	tags	views	score	done	label
0	Using entries from other kivy classes	61881920	['python', 'python-3.x', 'kivy']	12	0	False	python
1	Package python software with pylucene dependency	61896481	['python', 'docker', 'pip', 'dependencies', 'p...	7	1	False	python
2	Extracting time with regex from a string	61894597	['python', 'regex']	29	3	False	python
3	How do I add specific headers before each form...	61896721	['python', 'django']	4	0	False	python
4	Barplot from a dataframe using a column to set...	61896506	['python', 'pandas', 'bar-chart', 'seaborn', '...	12	0	True	python
...	...	...	...	...	...	...	...
75329	Php Monolog udp SocketHandler packet size	62436746	['php', 'udp', 'logstash', 'monolog']	16	1	False	php
75330	PHP CURL issue I Header Context Type is not se...	62435600	['php', 'curl']	11	0	False	php
75331	How can I delete data of single row in PHP dyn...	62436423	['php', 'html']	15	-3	False	php
75332	How to make custom associative array using an ...	62435957	['php', 'arrays', 'laravel', 'associative-array']	29	0	True	php
75333	base64 image to imagecreatefromstring() losing...	42385529	['javascript', 'php', 'base64', 'filereader', ...	817	1	True	php

75334 rows × 7 columns

Рис. 8. Stackoverflow Question Classification Challenge датасет

Перенося это на прикладной пример, обученная модель внутри системы чат-бота сможет определять подразделение внутри компании, к которому относится вопрос, или команду, внутри подразделения, и, с помощью отдельного запроса к базе данных или использования внутреннего справочника, отображать вспомогательную информацию по классу запроса (название подразделения, категория запроса, список ответственных лиц, список сотрудников, причастных к теме запроса и т.д.). Классификация запроса также позволяет анализировать намерения пользователя и тональность (эмоциональную окраску) сообщения. Данная информация может быть полезна для использования только необходимых модулей при решении задачи или формировать приоритетность пользователей исходя из их настроения или темы обращения. Все это значительно оптимизирует работу чат-бота, позволяя добиваться наивысшего качества при наименьших вычислительных и временных затратах.

Для решения задачи была выбрана «сырая» модель DistilBERT. Выбор был сделан с целью описания процесса дообучения классических BERT-моделей, а также демонстрации тонкой настройки сети исключительно на персональном наборе данных.

Выбранный набор данных представляет следующее распределение по классам (Рисунок 9).

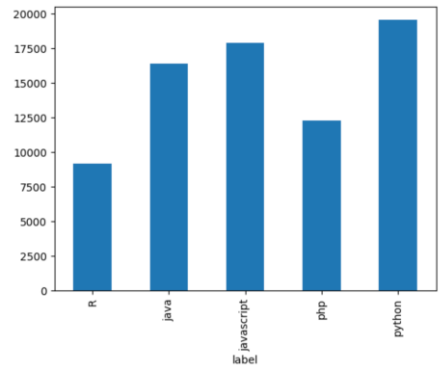


Рис. 9. Распределение классов в датасете

Поскольку выходом модели BERT является скрытый слой векторного представления входной последовательности, для решения задачи классификации необходимым минимум является добавление одного линейного слоя [31], который снижает размерность с 768 (стандартное значение размерности для BERT моделей [32]) до 5 (количество классов в персональном наборе данных).

Предварительно были проведены исследования, направленные на поиск оптимальной архитектуры модели. Рассматривалось влияние дополнительных линейный и дропаут слоев на тренировочную и валидацию точность. Схемы архитектур представлены на Рисунке 10а, 10б, 10в, 10г.

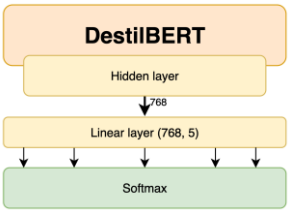


Рис 10а. Минимальная архитектура

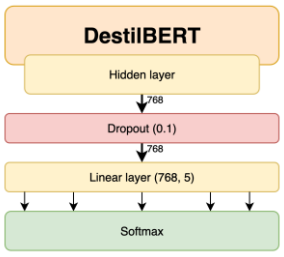


Рис 10б. Дополнительный слой Dropout

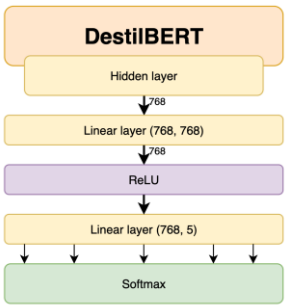


Рис 10в. Дополнительный линейный слой

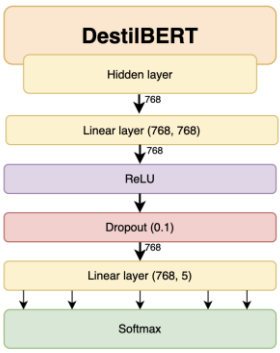


Рис 10г. Дополнительный линейный слой и слой Dropout

Для каждой архитектуры было запущено обучение на датасете, представляющем нормализированный по классам набор из двух тысяч случайных записей Stackoverflow Question Classification Challenge с параметрами, представленными в Таблице 4.

Таблица 4 — Параметры обучения для определения оптимальной архитектуры

	Оптимизатор	**Learning rate**
Параметры	Adam	1E-06

Результаты обучения на двадцати эпохах представлены на Рисунке 11.

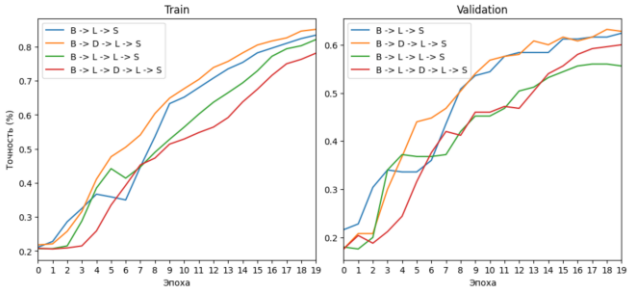


Рис. 11. Двадцать эпох обучения для разных архитектур

Из результатов видно, что хуже всего себя показали модели, включающие два линейных слоя. Это связано с тем, что избыточная сложность негативно влияет на семантическую плотность эмбеддингов, которой обладает предобученная модель DestilBERT. Модели, включающие лишь один линейный пуллинг слой, показали наилучшие результаты. Поскольку добавление дропаут

слоя в данный тип архитектуры также благоприятно повлиял на тренировочную и валидационную точность, было выбрано использовать данный тип модели для дальнейших испытаний, направленных на поиск оптимальный гиперпараметров и оптимизаторов.

После выбора оптимальной архитектуры модели была проведена нормализация данных (Рисунок 12). Из исходного датасета случайным образом было выбрано по две тысячи записей на каждый класс. Получившийся набор из десяти тысяч записей был использован для определения наиболее оптимальной архитектуры модели и гиперпараметров обучения.

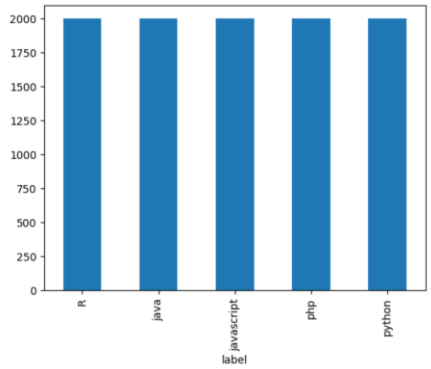


Рис. 12. Распределение классов после нормирования для тестового набора данных

В качестве функции активации была использована функция Softmax для получения вероятностных распределений по классам. В качестве функции потерь была выбрана CrossEntropyLoss, которая является стандартным выбором для многоклассовой взаимоисключающей классификации. Также был применен параметр label smoothing со значением 0.1, который слегка сглаживает целевые метки, уменьшая уверенность предсказаний модели. Данный подход позволил немного повысить устойчивость к ошибкам и уменьшил склонность модели к переобучению.

Модель обучалась с использованием оптимизаторов Adam, AdamW, SGD, RMSprop, Adadelta и Adagrad. Оптимизаторы Adadelta и Adagrad показали самые худшие результаты и неспособность к обучению. Для остальных вариантов в ходе прогонов пяти эпох на тестовом наборе данных были подобраны наиболее оптимальные гиперпараметры. Результат представлен в Таблице 5.

Таблица 5 — Оптимальные гиперпараметры для обучения

	Learning rate	**Weight decay**
Adam	1E-06	-
AdamW	5E-06	1E-04
SGD	5E-03	1E-04
RMSprop	1E-05	5E-05

Параметр weight decay был использован в качестве L2 регуляризации для предотвращения переобучения. Параметр BATCH SIZE был равен 16. Параметр максимальной длины последовательности был равен 64 и определен посредством анализа длин запроса в наборе данных (Рисунок 13).

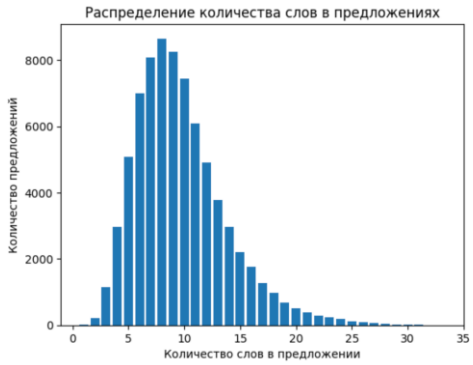


Рис. 13. Распределения количества слов в датасете Stackoverflow Question Classification Challenge

Результаты обучения на тестовом наборе данных для каждого из оптимизаторов представлены на Рисунке 14.

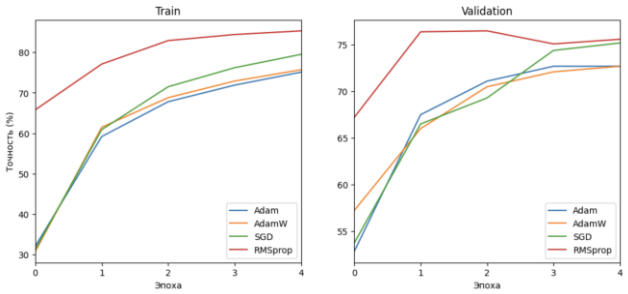


Рис. 14. Результаты обучения на тестовом наборе данных

Анализируя результаты обучения на тестовом наборе данных после пяти эпох, можно выделить оптимизатор RMSprop, который добивается наивысших результатов точности при меньшем времени обучения, однако демонстрирует склонность к переобучению. Хорошие результаты как на тренировочном, так и на валидационном наборе также демонстрирует оптимизатор SGD.

После тестовых запусков был сформирован итоговый набор данных, представляющий нормализированную версию исходного датасета. Распределение по классам представлено на Рисунке 15. Итоговый набор включал 45925 записей

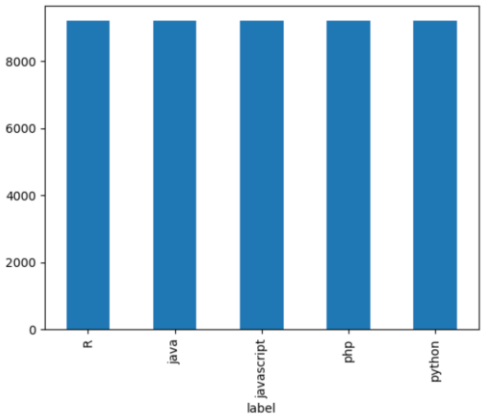


Рис. 15. Распределения классов итогового набора данных

Результаты обучения на итоговом наборе данных для каждого из оптимизаторов представлены на Рисунке 16.

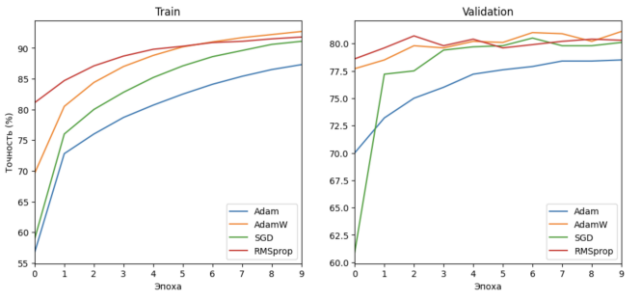


Рис. 16. Результаты обучения на итоговом наборе данных

Результаты обучения на итоговом наборе данных с увеличенным количеством эпох в первую очередь демонстрируют склонность к переобучению у модели во время использования оптимизаторов AdamW, SGD и RMSprop. Более стабильные, однако низкие результаты показывает оптимизатор Adam. Можно также сделать вывод в отсутствии необходимости увеличения количества эпох при увеличении набора данных.

Демонстрация точности на валидационных данных более восьмидесяти процентов при классификации коротких однотипных вопросов является хорошим показателем и подтверждает, что применение BERT в качестве основы для построения классификаторов позволяет достичь значительных успехов при минимизации временных и вычислительных затрат. При этом простота архитектуры снижает порог входа в решения задач обработки естественного языка, а высокая степень переносимости, свойственная предобученным моделям, позволяет успешно адаптировать архитектуру к самым разнообразным задачам.

IX. Валидация полученных моделей

После поиска оптимальной модели для семантического поиска и обучения модели для решения задачи классификации, был использован персональный набор данных, представляющий возможную базу знаний компании, для валидации итоговых алгоритмов и моделей

Распределение слов в данном наборе представлено на Рисунке 17. Набор также является нормализованным по классам

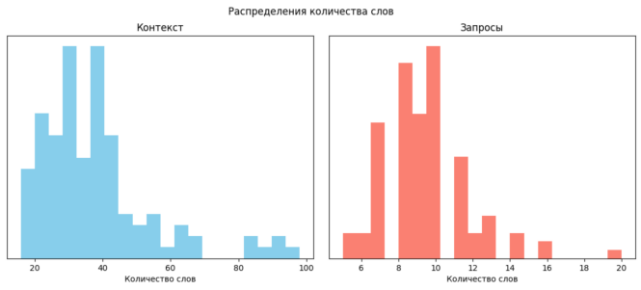


Рис. 17. Распределения количества слов для персонального набора данных

Для алгоритма поиска ближайших соседей была использовано модель Msmarco-distilbert-base-tas-b, которая показала наилучшие результаты в процессе исследований. Итоговая точность семантического поиска по группам из 1, 3 и 5 документов представлена в Таблице 6.

Таблица 6 — Результаты точности семантического поиска на персональном наборе данных

	Топ 1	Топ 3	Топ 5	Топ 10
Точность (%)	96.46	100	100	100

Для валидации модели классификации текста по 5 категориям были выбраны параметры, представленные в Таблице 7, которые показали наилучшие результаты в процессе исследований.

Таблица 7 — Параметры обучения модели, которая продемонстрировала наилучшую точность на тестовом наборе данных

	Оптимизатор	Learning rate	Weight decay
Параметры	AdamW	5E-06	1E-04

Результаты валидации модели на персональном наборе данных представлены на Рисунке 18.

```
Классификация текста
Персональный набор данных
Cross entropy loss, label smoothing: 0.1
AdamW, Learning rate: 5e-06, Weight decay: 1e-04
Точность:  90.27%
```

Рис. 18. Результаты точности обученной модели на персональном наборе данных при классификации запросов

Полученные валидационные данные на персональном датасете подтверждают высокую эффективность использования BERT-моделей для тонкой настройки, а также перспективу использования для построения архитектуры чат-бота, работающего на корпоративной базе знаний.

X. Архитектура чат-бота

Реализация алгоритма k-ближайших соседей, основанного на векторизации документов и запросов с помощью предобученных моделей DestilBERT, и тонкая настройка модели для классификации вопросов по категориям, которые были рассмотрены в данной статье уже позволяют построить простую архитектуру корпоративного чат-бота. Схема представлена на Рисунке 19.

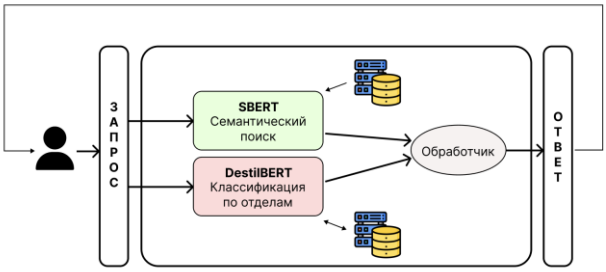


Рис. 19. Простая модель архитектуры чат-бота

В данном случае запрос пользователям обрабатывается независимости двумя нейронными сетям (подобный подход также может ускорить процесс с помощью параллельных вычислений). Результат выхода модели семантического поиска используется для определения наиболее вероятных статей. Данная информация нужна для формирования ранжированного списка с ссылками на документы. Выход модели-классификатора в данном случае определяет категорию запроса и с помощью об-

ращения к внутренним справочникам получает информацию об ответственных лицах и подразделениях. Пример работы данной связки представлен на Рисунке 20.

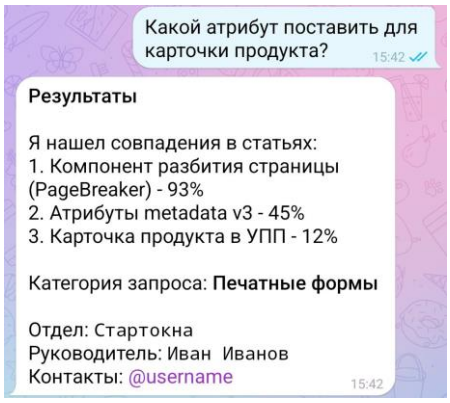


Рис. 20. Возможный пример работы простого чат-бота

Улучшить представленное решение можно переносом базы знаний в формат хранения документов, основанный на векторном представлении текста. Это снизит затраты на вычислительные ресурсы, а также увеличит скорость поиска по базе данных. Сам поиск может быть реализовон с помощью сервисов Faiss или Elasticsearch [33], [34].

Можно повысить качество ответов и оптимизировать вычислительные затраты, добавив в архитектуру чат бота дополнительные языковые модели, которые будут отвечать за классификацию намерений пользователя и генерацию текста. Упрощённый вариант новой архитектуры представлен на Рисунке 21.

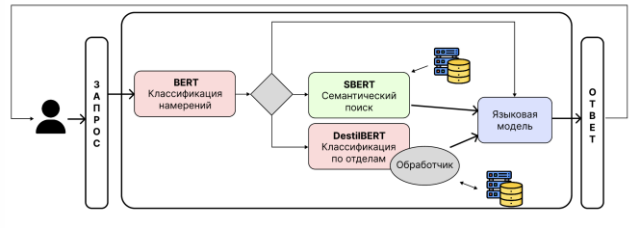


Рис. 21. Архитектура чат-бота с классификатором намерений и языковой моделью

Здесь модель классификатор определяет намерения пользователя. Функция обработки данных классификатора позволяет в случае необходимости подключать дополнительные модели для семантического поиска и классификации отделов компании. Выход формируется с помощью генеративной языковой модели (например, GPT) на основе полученных данных.

Более сложный вариант реализации содержит больше моделей, однако за счёт вспомогательных функций обработчиков в зависимости от запроса будут использоваться только необходимые для задачи нейронные сети, что не только снижает требование к вычислительным мощностям, но и оптимизирует затраты памяти, а учитывая, что над формированием ответа работают специально обученные под каждый тип nlp модели, данная архитектура будет также демонстрировать наивысшее качество. Схема представлена на Рисунке 22.

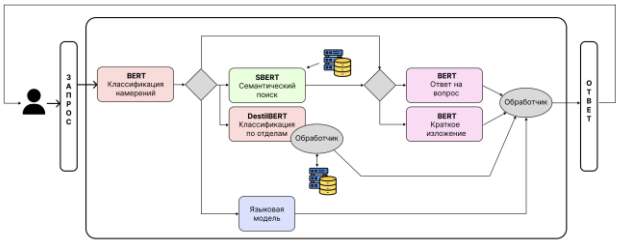


Рис. 22. Модульная архитектура чат-бота

Здесь обработка выхода классификатора намерений подключает только нужные для задачи модели. Это может быть лёгкая генеративная модель для поддержания диалога или связка семантического поиска с последующей обработкой результата моделью предназначенной для ответов на вопрос или краткого изложения

Выбор архитектуры исходит от требований компании к корпоративному чат боту и ресурсных ограничений серверов. Модульная же архитектура позволяет обеспечить гибкость и масштабируемость системы, а также добиться максимального качества ответов при низких ресурсных затратах

XI. Заключение

В ходе исследования, описанного в данной статье, был проведен анализ подходов к использованию предобученных моделей, таких как BERT, для решения комплексных задач в области обработки естественного языка (NLP). Практическое применение этих моделей показало их эффективность в задачах семантического поиска и классификации текста, а возможное использование BERT моделей для решения других типов задач вместе с генеративными языковыми моделями даёт возможность создание модульной архитектуры, которая покрывает все необходимые требования для разработки корпоративного чат бота и обеспечивает гибкость и масштабируемость системы.

Анализ преимуществ использования предобученных моделей подчеркнул их способность значительной экономии времени и ресурсов, поскольку они требуют минимальной дополнительной настройки для решения специфических задач, обеспечивая высокое качество. Во многих случаях использование таких моделей позволяет избежать длительного и ресурсоемкого процесса обучения с нуля, предоставляя обширное базовое понимание естественного языка, которое может быть адаптировано для конкретных нужд бизнеса.

Литература

1. N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.
2. A. A. Yakovlev, A. B. Kondybayeva and S. V. Solodov, "Intelligent System for Collecting, Analyzing and Managing Data in the Field of Medicine," 2019 Wave Electronics and its Application in Information and Telecommunication Systems (WECONF), St. Petersburg, Russia, 2019, pp. 1-6, doi: 10.1109/WECONF.2019.8840588.
3. A. N. Semochkin, S. Zabihifar and A. R. Efimov, "Object Grasping and Manipulating According to User-Defined Method Using Key-Points," *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, Kazan, Russia, 2019, pp. 454-459, doi: 10.1109/DeSE.2019.00089.

4. Smolin, A.; Yamaev, A.; Ingacheva, A.; Shevtsova, T.; Polevoy, D.; Chukalina, M.; Nikolaev, D.; Arlazarov, V. Reprojection-Based Numerical Measure of Robustness for CT Reconstruction Neural Network Algorithms. *Mathematics* 2022, *10*, 4210. https://doi.org/10.3390/math10224210.
5. Josh Brown, "Knowledge Base Guide: Why Your Business Needs One", *helpjuice*. Available at: https://helpjuice.com/blog/knowledge-base.
6. Elissaveta Gourova, "Knowledge management strategy for Small and Medium Enterprises". Conference: IEEE-AM ACS At: Malta Volume: pp. 639-648.
7. Ian Alton, "The knowledge base problem", *Bootcamp*. Available at: https://bootcamp.uxdesign.cc/the-knowledge-base-problem-bc379de7408f.
8. Yogesh Malhotra, "Why Knowledge Management Systems Fail? Enablers and Constraints of Knowledge Management in Human Enterprises". *ResearchGate*. DOI:10.1007/978-3-540-24746-3_30. Available at: https://www.researchgate.net/publication/228585526_Why_Knowledge_Management_Syste ms_Fail_Enablers_and_Constraints_of_Knowledge_Management_in_Human_Enterprises.
9. Linmei Hu, Zeyi Liu, Ziwang Zhao, Lei Hou, Liqiang Nie, Juanzi Li, "A Survey of Knowledge Enhanced Pre-trained Language Models". arXiv:2211.05994.
10. Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, Sebastian Riedel, "Language Models as Knowledge Bases?". arXiv:1909.01066.
11. Tech Desk, "How much does it cost to train custom GPT-4 model?", *TheIndianEXPRESS*. Available at: https://indianexpress.com/article/technology/artificial-intelligence/custom-gpt-4-model-training-cost-features-9019955/.
12. Vinayedula, "Top 5 Pre-Trained Models in Natural Language Processing", *DSA*. Available at: https://www.geeksforgeeks.org/top-5-pre-trained-models-in-natural-language-processing-nlp/.
13. Constantine Chung, "The Next LLMs Development. Mixture-of-Experts with Expert Choice Routing". Available at: https://hkaift.com/the-next-llms-development-mixture-of-experts-with-expert-choice-routing/.
14. Hugging Face, "DistilBERT". Available at: https://huggingface.co/docs/transformers/model_doc/distilbert.
15. R. Dale, "Classical approaches to natural language processing". *ResearchGate*. Available at: https://www.researchgate.net/publication/328305088_Classical_approaches_to_natural_lang uage_processing.
16. Pratyush Khare, "Deep Learning for NLP: Word2Vec, Doc2Vec, and Top2Vec Demystified", *Medium*. Available at: https://medium.com/mlearning-ai/deep-learning-for-nlp-word2vec-doc2vec-and-top2vec-demystified-3842b4fad5c9.
17. Xipeng Qiu, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, Xuanjing Huang. «Pre-trained Models for Natural Language Processing». *A Survey*. arXiv:2003.08271.
18. Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin, "Attention Is All You Need". arXiv:1706.03762.
19. Jay Alammar, "Visualizing A Neural Machine Translation Model (Mechanics of Seq2seq Models With Attention)". Available at: https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/.
20. Jay Allamar, "The Illustrated Transformer". Available at: https://jalammar.github.io/illustrated-transformer/.
21. Niklas Heidloff, "Foundation Models, Transformers, BERT and GPT". Available at: https://heidloff.net/article/foundation-models-transformers-bert-and-gpt/.
22. Яков Длугач, "Не GPT единым: преимущества BERT перед ChatGPT". Available at: https://rb.ru/opinion/bert-vs-gpt/.
23. Fernando Aguilar, Chris Marino, "Expert Analysis: Keyword Search vs Semantic Search – Part One". Available at: https://enterprise-knowledge.com/expert-analysis-keyword-search-vs-semantic-search-part-one/.
24. Skillfactory, "Семантический поиск: от простого сходства Жаккара к сложному SBERT". Available at: https://habr.com/ru/companies/skillfactory/articles/566414/.
25. OTUS, "Извлечение признаков из текстовых данных с использованием TF-IDF". Available at: https://habr.com/ru/companies/otus/articles/755772/.
26. Subir Verma, "Semantic Search with S-BERT is all you need", *Medium*. Available at: https://medium.com/mlearning-ai/semantic-search-with-s-bert-is-all-you-need-951bc710e160.
27. Юрий Басаров, "Как с помощью BERT организовать поиск похожих текстов". Available at: https://habr.com/ru/articles/682630/.
28. Antony Christopher, "K-Nearest Neighbor". Available at: https://medium.com/swlh/k-nearest-neighbor-ca2593d7a3c4.
29. Vijaya Sasidhar Nagella, "Cosine Similarity Vs Euclidean Distance". Available at: https://medium.com/@sasi24/cosine-similarity-vs-euclidean-distance-e5d9a9375fc8'.
30. Sentence-Transformers, "Semantic Search". Available at: https://www.sbert.net/examples/applications/semantic-search/README.html.
31. NTA, "Классификация текста с использованием моделей трансформеров". Available at: https://newtechaudit.ru/klassifikacziya-teksta-s-ispolzovaniem-modelej-transformerov/.
32. Hugging Face, "Pretrained models". Available at: https://huggingface.co/transformers/v2.10.0/pretrained_models.html.
33. Scapper, "Семантика в масштабе: BERT + Elasticsearch", *MachineLearningMastery.ru*. Available at: https://machinelearningmastery.ru/semantics-at-scale-bert-elasticsearch-be5bce877859/.
34. Ioannis Tsiokos, "Add Similarity Search to DynamoDB with Faiss". Available at: https://medium.com/swlh/add-similarity-search-to-dynamodb-with-faiss-c68eb6a48b08.

Классификация последовательных текстовых данных с использованием архитектуры LSTM на основе квантовой схемы

А.А. Виговский
кафедра инженерной кибернетики
НИТУ МИСИС
Москва, Россия
aavigovskij@gmail.com

***Аннотация* — данная статья посвящена теме развития способов применения квантовых методов в области нейронных сетей в общем и в области обработки последовательных данных, в частности. В данной статье производится обзор технологии использования квантовых схем в нейронных сетях на основе архитектуры LSTM, а также применяется гибридная квантово-классическая нейронная сеть для обработки данных из двух различных наборов для решения задачи двоичной классификации данных. Также в статье производится доработка классической нейронной сети с помощью квантовой схемы, имитирующей поведение блока LSTM.**

Keywords—LSTM, нейронные сети, квантовые вычисления, гибридные квантово-классические нейронные сети, вариационные квантовые схемы

I. Введение

В настоящее время производится большое количество исследований будущих возможностей применения квантовых технологий, которые показывают, что они позволяют повышать эффективность решения множества задач [1], как, например, алгоритм Гровера имеет меньшую вычислительную сложность для поиска в БД или алгоритм Шора, который позволяет решать задачу поиска простых множителей, что заставляет разработчиков изменять подходы к применению, например, криптографии в компьютерных сетях.

В то же время, огромной значимостью для науки [2] и других областей, имеют нейронные сети и искусственный интеллект. Алгоритмы машинного обучения и искусственного интеллекта применяются в таких областях как: беспилотный транспорт [3], компьютерное зрение и распознавание объектов [4], распознавание текста [5], а также являются основой для попытки разработки общего искусственного интеллекта [6].

Данные обстоятельства указывают на важность развития и применения сферы квантовых вычислений во всех областях применения информационных технологий, в особенности, в развитии нейронных сетей и искусственного интеллекта.

Однако в рамках данной статьи уделяется внимание применению вариационных квантовых схем [7] в области построения нейронных сетей на основе архитектуры LSTM [8] и ее применение в решении задачи бинарной классификации последовательных данных. Основными направлениями, в которых возможно применение вариационных квантовых нейронных схем являются полные квантовые нейронные сети, а также гибридные квантово-классические нейронные сети.

II. Использование квантовых методов в области нейронных сетей

Текущая эпоха развития квантовых компьютеров называется Noisy intermediate-scale quantum era [9], что означает эпоху, которая характеризуется тем, что квантовые процессоры могут содержать до 1000 кубит, что уже позволяет достигать квантового превосходства в некоторых алгоритмах, но недостаточны для повсеместного решения задач классических алгоритмов квантовыми компьютерами.

Так, проводились исследования, в которых происходило сравнение эффективности обучения нейронной сети, целиком основанной на квантовой схеме, и нейронной сети, основанной на классических практиках построения нейронных сетей, который показал, что хоть нейронной сети на основе квантовой схемы потребовалось значительно меньшее количество параметров и эпох на обучение, для того, чтобы квантовая нейронная сеть могла обработать данные, размерность набора данных MNIST необходимо было кардинально уменьшить, с 784 пикселей до всего лишь 16 пикселей, так как для большей размерности данных, было необходимо большее, недоступное количество кубитов. Данное исследование показывает, что хоть целиком квантовые нейронные сети и справляются с решением тех же задач лучше, чем классические нейронные сети, на данном этапе развития квантовых вычислителей, они не могут решать те же задачи, что и классические нейронные сети.

В качестве решения данной проблемы, был предложен подход, который можно назвать как гибридная квантово-классическая нейронная сеть [10].

III. Применение LSTM архитектуры

A. LSTM в классических нейронных сетях

Для того, чтобы разобраться в том, каким образом могут быть использованы гибридные квантово-классические нейронные сети на основе архитектуры LSTM, необходимо разобраться в том, на чем основана классическая архитектура LSTM, а именно, в идее реккурентной нейронной сети [11].

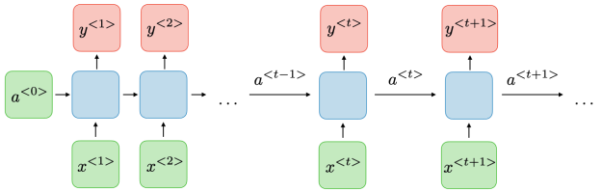


Рисунок 1. Пример общей архитектуры реккуретной нейронной сети

На приведенном выше рисунке N показано, что каждый слой сети имеет не только вход x^i, но и вход a^t, который позволяет учитывать не только текущие входные данные, но и результат работы над предыдущими данными.

Такая организация сети позволяет обрабатывать последовательные данные, в которых предполагается необходимость учитывать контекст текущей порции данных.

Такие сети имеют две существенные проблемы, называемые проблемами исчезающего и взрывающегося градиентов [12].

Архитектура LSTM является логическим продолжением идеи реккруретных нейронных сетей, особенность которых заключается в том, что они позволяют обрабатывать последовательные данные за счет построения сети таким образом, что входом слоя сети являются не только новые данные, но и выходные данные со слоя обработки предыдущих данных. LSTM позволяет воссоздать реккуретную архитектуру для работы с последовательными данными, которая не будет подвержена проблемам исчезающего и взрывающихся градиентов.

Для решения обозначенных проблем, LSTM использует две функции активации: сигмоиду и гиперболический тангенс, что позволяет реализовать несколько гейтов, через которые должны пройти данные для корректной обработки.

Использование сигмоиды в качестве функции активации обусловлено множеством значений данной функции, которое представляет из себя интервал между значениями 0 и 1, что позволяет определять процент данных, сохраняемых на различных слоях LSTM архитектуры.

Приведем схему классической LSTM архитектуры на рисунке 2 с обозначенными гейтами [13].

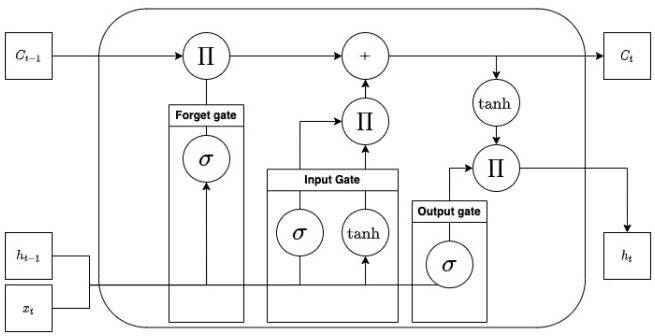


Рисунок 2. Архитектура LSTM с обозначением гейтов

Обозначим входные параметры модуля LSTM:

- $x(t)$ – вектор обрабатываемых на данном шаге данных.
- c_{t-1} – вектор, называемый состоянием ячейки, который может быть ассоциирован с долговременной памятью модели. На пути следования данного вектора по ячейке отсутсвуют какие-либо веса, что помогает решить проблему исчезающего и взрывающегося градиентов, свойственной для классической архитектуры реккурентных нейронных сетей.
- $h(t-1)$ – вектор данных, который может быть ассоциирован с краткосрочной памятью модели. В отличие от состояния ячейки, перед подачей данных на гейты, значение вектора подвергается воздействию весов, вычисляемых нейронной сетью в ходе обучения.

Также на схеме представлены гейты – набор компонентов, преобразовывающих входные данные системы. На схеме выделено 3 таких схемы, опишем более подробно, за что отвечает каждый из гейтов:

- Forget gate – гейт, определяющий, какое количество информации, полученной при обработке предыдущих данных должно быть отброшено. Ключевую роль в данном процессе играет функция сигмоиды, множеством значений которой является диапазон значений от 0 до 1, что позволяет получить процент информации, который должен быть отброшен.
- Input gate – гейт, который определяет на основе входной информации и кратковременной памяти (значения h_{t-1}), какой процент входной информации будет добавлен к состоянию ячейки (сохранен в качестве долговременной памяти). Выход данного гейта влияет на значение C_t –
- Output gate – гейт, который определяет, какой процнт входной информации стоит сохранить в качестве кратковременной памяти, так как выходное значение данного гейта направляется в качестве выхода h_t всего LSTM-модуля.

B. LSTM на основе квантовой схемы

К области квантовых вычислений используются вариационные квантовые схемы – специальный класс квантовых схем, характерной особенностью которого является тот факт, что они могут быть параметризированы, что может быть реализовано с помощью операторов поворота. Оператор поворота – такой оператор, воздействующий на кубит, который может привести его в любое состояние, зависящее от угла поворота. В вариационных квантовых схемах параметром является угол поворота кубита, что позволяет производить обучение квантовой схемы так, что в классических нейронных сетях, параметрами выступают векторы действительных чисел, а в вариационных квантовых схемах в качестве обучаемых параметров выступают углы поворота кубита относительно различных осей.

Для визуализации поворота состояния кубита в квантовых вычислениях используется сфера Блоха [14], на которой обозначены векторы состояний и углы поворота данных векторов относительно различных осей.

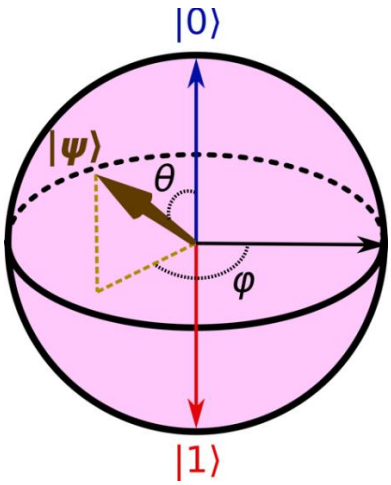


Рисунок 3. Визуализация сферы Блоха

Простейшим примером вариационной квантовой схемы может служить аппроксимация оператора Паули X, который воздействует на кубит таким образом, что из состояния $|0\rangle$ он переходит в состояние $|1\rangle$, и наоборот. В процессе обучения данной сети, параметром будет выступать значение угла θ, а в ходе обучения, схема будет производить поворот на угол, и в случае несовпадения состояния с ожиданием, будет увеличивать значение угла на некоторый параметр. Таким образом, когда состояние угла θ достигнет значения, близкого к значению π, обучение схемы может быть остановлено. Это очень упрощенный пример с одним параметром, однако он наглядно демонстрирует схожесть вариационных квантовых схем и классических нейронных сетей.

Рассмотрим, как может быть реализована квантовая схема на основе архитектуры LSTM

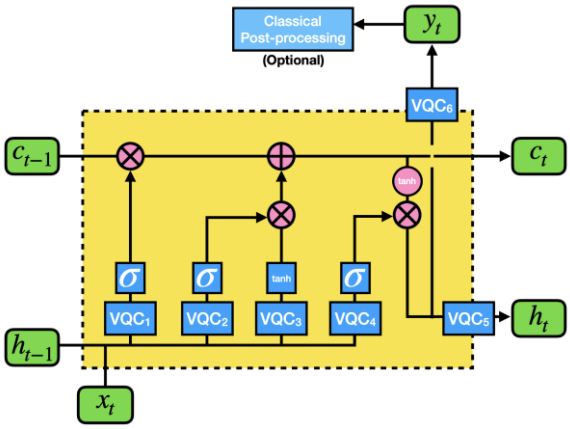


Рисунок 4. Архитектура LSTM на основе вариационной квантовой схемы

Как можно видеть, из схемы, вариационные квантовые схемы [15], обозначаемые как VQC_i, в данном случае, являются параметризируемым слоем, который является предметом обучения нейронной сети.

Приведем пример вариационной квантовой схемы, используемой в архитектуре LSTM.

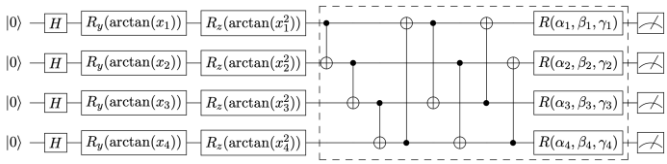


Рис. 5. Общая схема VQC, используемой в архитектуре LSTM

Как можно видеть, на данной схеме присутствуют параметризованные операторы поворота состояния кубита, которые являются предметом обучения. Таким образом, предметом обучения и оптимизации являются значения углов $\alpha_1, \beta_1, \gamma_1$, соответственно являющихся углами поворота вектора состояния кубита относительно осей x, y, z.

Рассмотрим, какое изменение производит с представленными кубитами в ходе выполнения квантовой схемы:

1. Применение оператора Адамара – оператора, приводящего кубит из состояния $|0\rangle$ в состояние равновесных амплитуд, что означает, что после применения данного оператора, при измерении состояния кубита, с равной вероятностью будет получено состояние 0 или 1.
2. Следующим шагом происходит вращение вектора состояния кубита относительно оси y на значение угла, равное значению $arctan(x_i)$ и $arctan(x_i^2)$, где x_i – значения входного вектора.
3. Далее происходит последовательное применение операторов CNOT, используемых для того, чтобы привести кубиты в состояние квантовой запутанности [16].
4. После перевода кубитов в состояние квантовой запутанности, происходит параметризированное изменение состояний кубитов путем вращения вектора состояния кубита на уже упомянутые ранее углы α, β, γ.
5. В конце выполнения вариационной квантовой схемы, находится операция измерения состояний кубитов. Данная операция может быть выполнена как на квантовом компьютере, так и на классическом компьютере, на котором установлен симулятор квантового компьютера, в данной работе будет использована библиотека Pennylane [17].

IV. Наборы данных

A. *IMDB of 50K Movie Review*

Набор представляет из себя совокупность рецензий к фильмам на одном из главных порталов оценки кинофильмов – IMDB. Данный набор представляет из себя большое количество классифицированных записей, в которых каждый отзыв разделен на два класса – positive и negative, что позволяет производить обучение нейронных сетей на задачах классификации текстовых данных [18].

Задача по обработке этого набора данных состоит в том, чтобы обучить нейронную сеть классифицировать новые отзывы на принадлежность к классам, основываясь

на параметрах, полученных в ходе обучения на рассматриваемом наборе данных.

Рассмотрим, какой объем данных представлен в рассматриваемом наборе. Подмножество данных для тренировки модели составляет 75% от всего набора данных или 37500 из 50 000, подмножество данных для валидации модели, в свою очередь, составляет 25% или 12 500 из 50 000.

Каждый отзыв представляет из себя символьную последовательность различной длины, на рисунке 6 приведена диаграмма, показывающая количество данных в диапазонах длин отзывов.

Для того, чтобы нейронная сеть могла обрабатывать текстовые данные, необходимо преобразовать их в численный вид. Данный процесс может быть выполнен при помощи преобразования неслучайной символьной последовательности в векторный вид. Как правило, слой, решающий данную задачу, называется embedding-слой.

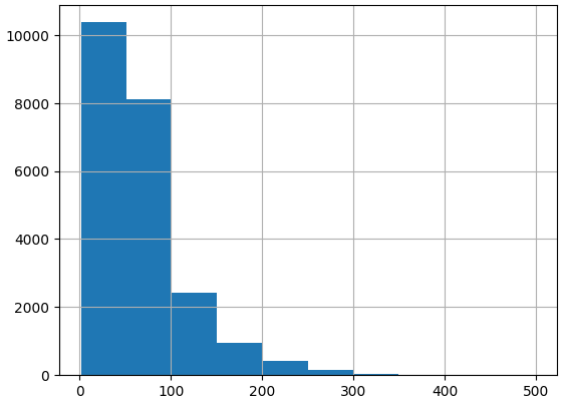


Рисунок 6. Столбчатая диаграмма количества отзывов по длине

B. *Natural Language Processing with Disaster Tweets*

Набор данных представляет из себя набор двоично классифицированных текстовых данных из социальной сети Twitter. Каждое сообщение представляет из себя текст сообщения, а также класс, где метка 0 означает принадлежность сообщения к классу сообщений, не относящихся к чрезвычайному происшествию, а 1, напротив, означает, что сообщение описывает чрезвычайное происшествие.

Для данного набора данных приведем частоту встречи слова в наборе для нескольких наиболее встречаемых слов:

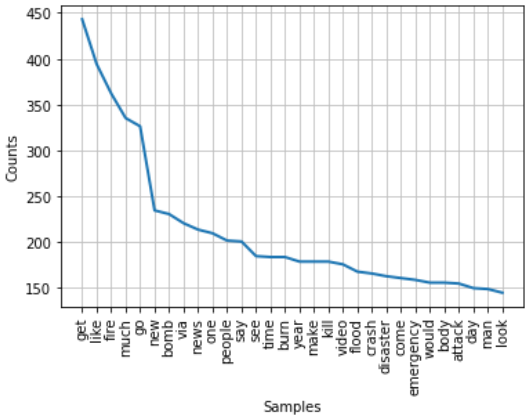


Рисунок 7. Сопоставление слова и количества его появлений среди записей с классом 1

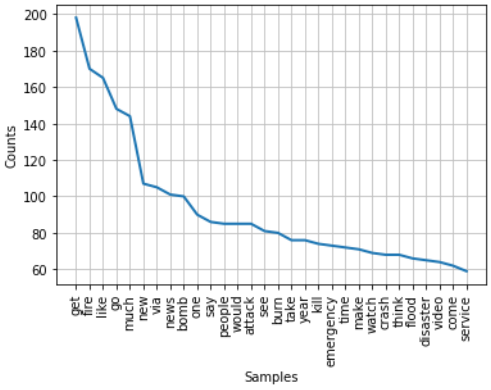


Рисунок 8. Сопоставление слова и количества его появлений среди записей с классом 0

V. Проведенные исследования и результаты

Для того, чтобы рассмотреть преимущества использования вариационных квантовых схем в сравнении с классическим подходом, предлагается провести сравнительный анализ нейронных сетей с двумя подходами следующим образом: провести обучение классической нейронной сети (без использования вариационных квантовых схем) на приведенных ранее наборах данных, а также провести обучение гибридной квантово-классической нейронной сети [19] на тех же наборах данных, после чего, сравнить показатели обучения полученных нейронных сетей.

Для сравнения результатов предлагается сравнить показатели точности и значений функций потерь для обеих реализаций нейронной сети.

A. *Получение гибридной квантово-классической из классической сети на основе архитектуры LSTM*

В качестве задач, для решения которых будет использованы нейронные сети на основе архитектуры LSTM, выбраны описанные ранее задачи бинарной классификации текстовых данных.

Для решения данных задач были взяты решения, в которых используется фреймворк PyTorch [20], а также реализация слоя LSTM на основе вариационной квантовой схемы. Суть реализации гибридной сети заключается в том, чтобы заменить в реализации

классической нейронной сети, слой LSTM на класс из библиотеки qlearnkit, который является реализацией вариационной квантовой схемы. Так как данный класс реализован с использованием фреймворка PyTorch, для такой реализации необходимо и достаточно, чтобы данный класс являлся имплементацией класса Module фреймворка PyTorch.

Для того, чтобы проверить работоспособность данного решения, было проведено обучение нейронной сети, решать поставленные в описании наборов данных задачи.

За счет модульности архитектуры ПО на основе PyTorch, данная архитектура нейронной сети позволяет гибко изменять количество слоев и параметров сети под различные задачи, а также при необходимости вносить изменения и в саму архитектуру решения.

Для проведения анализа, обучение будет производиться для двух нейронных сетях: сети, в которой LSTM-модуль был реализован с использованием модуля из библиотеки PyTorch, и сети, в которой используется модуль из библиотеки qlearnkit с использованием схемы из 4 кубитов.

B. Обучение нейронных сетей

Произведем обучение нейронных сетей на представленных наборах данных. Для исследования эффективности обучения нейронной сети, используются различные параметры. Так, квантовая схема позволяет задавать количество кубитов, используемых в ходе обучения схемы.

В виду недоступности реального квантового компьютера, предлагается использование симулятора квантового компьютера, доступного за счет использования библиотеки Pennylane. В силу того, что используется симулятор, квантовые вычисления не получат скорости за счет аппаратного ускорения операций, однако алгоритмы, выполняемые симулятором, будут иметь вычислительную сложность, соответствующую квантовым алгоритмам.

В ходе выполнения обучения нейронной сети, предлагается выполнить по 15 эпох обучения сети на каждый набор данных.

В качестве результатов обучения приведем графики зависимости показателя точности и ошибок в зависимости от эпохи. На каждом графике будут приведены данные показатели как для классической реализации нейронной сети, так и для гибридной реализации на основе квантовой схемы.

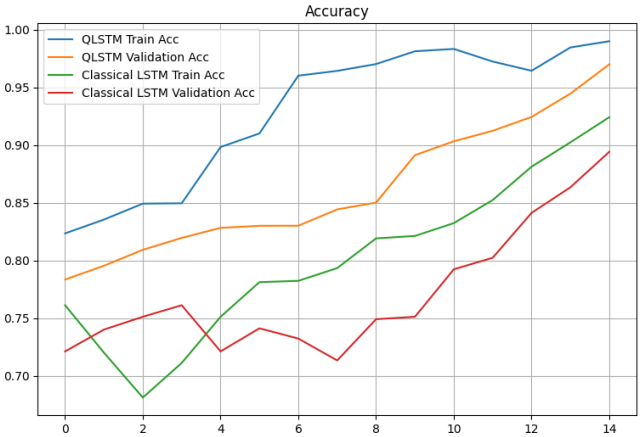


Рисунок 9. Зависимость значений точности для набора данных IMDB

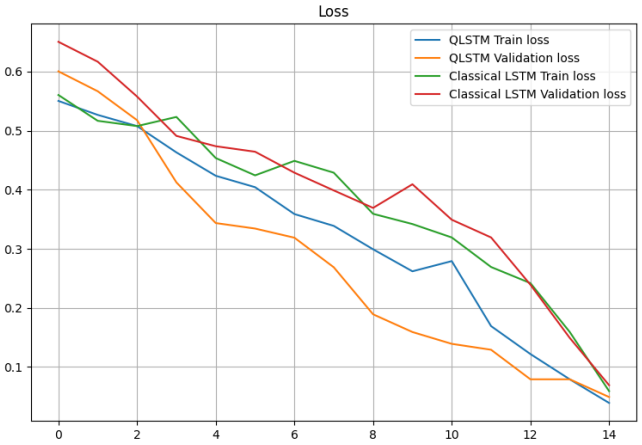


Рисунок 10. Зависимость функции потерь от количества эпох для набора данных IMDB

Приведенные графики показывают зависимость параметров, используемых для проверки работоспособности нейронной сети от количества эпох. На основе данных графиков можно видеть о последовательном повышении точности нейронной сети, и снижении показателей функции потерь по мере увеличения количества эпох. Данные выводы справедливы для обеих архитектур нейронных сетей. Также следует отметить, что при последующем увеличении количества эпох, нейронная сеть начинает показывать результаты меньшего качества на валидационном наборе данных [21], что может говорить о переобучении [22] нейронной сети. Предотвратить данную проблему возможно эмпирическим путем, проводя большее количество испытаний на большем количестве наборов данных.

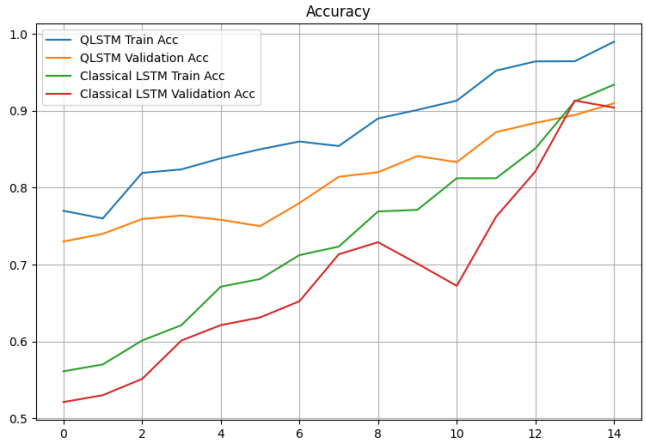


Рисунок 11. Зависимость значения точности от количества эпох для набора данных Disaster Tweets

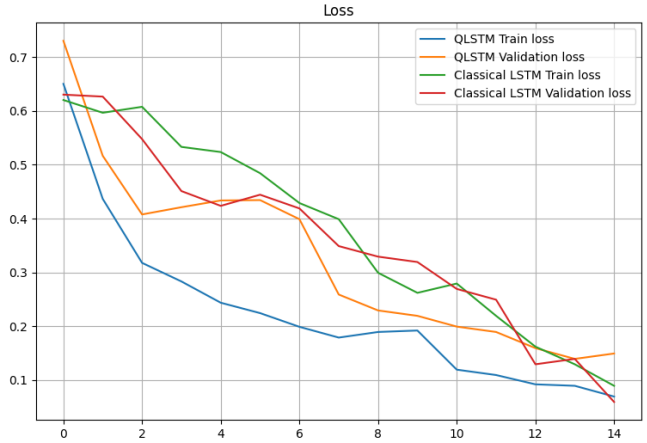


Рисунок 12. Зависимость функции потерь от количества эпох для набора данных Disaster Tweets

C. Сравнение результатов

Рассмотрим результаты, которые были показаны обеими нейронными сетями за отведенное количество эпох. На каждом рисунке можно отметить, что графики приведенных параметров сводятся к значениям 0 (для точностей нейронных сетей), а также к значению 1 (для значений функции потерь).

Однако следует отметить, каким образом сети движутся к обозначенным результатам. Так, результаты обучения нейронных сетей на основе квантовых схем, показывают меньшее количество выбросов, а также, для обоих наборов данных, нейронным сетям на основе квантовых схем необходимо меньшее количество эпох для достижения тех же показателей, что и для классических нейронных сетей.

Так, для набора данных IMDB, квантовые нейронные сети, преодолевают значение точности 0.9 для тренировочного набора данных уже на 4-й эпохе обучения, в то время как для классической нейронной сети, такой показатель достигается только на 13 эпохе. Та же динамика сохраняется и для другого набора данных. Данное свойство квантовых схем в нейронных сетях может быть использовано для оптимизации процесса обучения на более сложных данных, особенно, учитывая, что в данной работе была использована только четырехкубитная квантовая схема, тогда как в условиях доступности большего количества кубитов, результат может оказаться еще более значимым.

Следует отметить, что данные выводы были получены и для других исследований и архитектур.

VI. Заключение

В процессе выполнения работы было произведено обучение нейронной сети на основе архитектуры LSTM с квантовым LSTM модулем, а также обучение схожей по архитектуре сети на основе классического подхода LSTM, для реализации которых был использован фреймворк PyTorch, что позволило создать гибкие повторно используемые модели для решения задач различных наборов данных.

Была рассмотрена идея использования четырехкубитных вариационных квантовых схем в моделях нейронных сетей, что позволяет получить преимущества квантовых вычислений, а также избежать издержек, появляющихся при использовании квантовых схем эры NISQ. Также было проведено сравнение показателей нейронных сетей с различной реализацией LSTM-модулей, на квантовой схеме и классической.

В качестве дальнейшего развития данной темы, возможен запуск обучения данных моделей с использованием квантового компьютера [23] и сравнения количества времени, затраченного на исполнение разработанного кода. Запуск данных квантовых схем на квантовом компьютере позволил бы использовать большее количество кубитов, что позволило производить более качественное и быстрое обучение вариационной квантовой схемы, лежащей в основе данной нейронной сети [24].

Литература

[1] Yung, Man-Hong. (2019). Quantum supremacy: Some fundamental concepts. National Science Review. 6. 22-23. 10.1093/nsr/nwy072.

[2] Anokhin, K. V., for Advanced Brain Studies, I., Novoselov, K. S., Smirnov, S. K., Efimov, A. R., Matveev, P. M., … University, L. M. S. AI for Science and Science for AI. Voprosy Filosofii, 93–105.

[3] Ali, Bushra & Sadekov, Rinat. (2023). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation. 30. 87–105. 10.17285/0869-7035.00105.

[4] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[5] Polevoy, Dmitriy & Kulagin, Petr & Ingacheva, Anastasia & Soldatova, Zhanna & Chukalina, Marina & Nikolaev, Dmitriy & Arlazarov, Vladimir. (2023). From tomographic reconstruction to automatic text recognition: the next frontier task for the artificial intelligence. 51. 10.1117/12.2680132.

[6] Efimov, A.R. & Dubrovsky, D.I. & Matveev, P.M. (2023) What Prevents Us from Creating Artificial General Intelligence? One Old Wall and One Old Dispute. Voprosy Filosofii, 39–49.

[7] Karafyllidis, Ioannis. (2005). Quantum Computer Simulator Based on the Circuit Model of Quantum Computation. Circuits and Systems I: Regular Papers, IEEE Transactions on. 52. 1590 - 1596. 10.1109/TCSI.2005.851999.

[8] Staudemeyer, Ralf & Morris, Eric. (2019). Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks.

[9] Preskill, John. (2018). Quantum Computing in the NISQ era and beyond. Quantum. 2. 10.22331/q-2018-08-06-79.

[10] Arthur, Davis & Date, Prasanna. (2022). A Hybrid Quantum-Classical Neural Network Architecture for Binary Classification.

[11] Du, Ke-Lin & Swamy, M.N.s. (2014). Recurrent Neural Networks. 10.1007/978-1-4471-5571-3_11.

[12] Hochreiter, Sepp. (1998). The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems. 6. 107-116. 10.1142/S0218488598000094.

[13] Samuel Yen-Chi Chen, Shinjae Yoo, and Yao-Lung L. Fang (2020). Quantum Long Short-Term Memory.

[14] Wharton, Ken B. and Denise M. Koch. "Unit quaternions and the Bloch sphere." Journal of Physics A: Mathematical and Theoretical 48 (2014): n. pag.

[15] Cerezo, Marco & Arrasmith, Andrew & Babbush, Ryan & Benjamin, Simon & Endo, Suguru & Fujii, Keisuke & Mcclean, Jarrod & Mitarai, Kosuke & Yuan, Xiao & Cincio, Lukasz & Coles, Patrick. (2021). Variational quantum algorithms. Nature Reviews Physics. 3. 1-20. 10.1038/s42254-021-00348-9.

[16] Ma, Hongbao & Young, Margaret & Yan, Yang. (2016). Quantum Entanglement Introduction. 8. 93-97. 10.7537/marsaaj080716.13.

[17] Bergholm, Ville & Izaac, Josh & Schuld, Maria & Gogolin, Christian & Killoran, Nathan. (2018). PennyLane: Automatic differentiation of hybrid quantum-classical computations.

[18] Tripathi, Sandesh & Mehrotra, Ritu & Bansal, Vidushi & Upadhyay, Shweta. (2020). Analyzing Sentiment using IMDb Dataset. 30-33. 10.1109/CICN49253.2020.9242570.

[19] Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, Rabab Ward Deep Sentence Embedding Using Long Short-Term Memory Networks: Analysis and Application to Information Retrieval.

[20] Ketkar, Nikhil. (2017). Introduction to PyTorch. 10.1007/978-1-4842-2766-4_12.

[21] Kakarash, Zana. (2023). Why is data validation important in research?. 10.13140/RG.2.2.34496.81920.

[22] Ying, Xue. (2019). An Overview of Overfitting and its Solutions. Journal of Physics: Conference Series. 1168. 022022. 10.1088/1742-6596/1168/2/022022.

[23] Dahi, Zakaria Abdelmoiz & Alba, Enrique & Gil-Merino, Rodrigo & Chicano, Francisco & Luque, Gabriel. (2021). A Survey on Quantum Computer Simulators.

[24] Bach, Bao & Kundu, Akash & Acharya, Tamal & Sarkar, Aritra. (2023). Visualizing Quantum Circuit Probability -- estimating computational action for quantum program synthesis.

Исследование возможности детектирования эмоций

А. Г. Лойко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2314262@edu.misis.ru

Аннотация — Современные технологии активно развиваются в области распознавания человеческих эмоций, что представляет особую ценность в таких сферах, как медицина, безопасность, маркетинг и интерактивные услуги. Распознавание эмоций лица является ключевой задачей, включающей идентификацию различных эмоциональных состояний, таких как радость, грусть, злость и удивление. Методы глубокого обучения, включая архитектуру AlexNet, показывают значительные успехи в этой области. В работе исследуются различные открытые решения и сравниваются их возможности по распознаванию эмоций на наборах данных AffectNet и Emotional Detection.

Ключевые слова — распознавание эмоций, глубокое обучение, анализ выражения лица, компьютерное зрение, AlexNet, AffectNet, Emotional Detection

I. Введение

Исследования и разработки в области распознавания эмоций по изображениям активно ведутся в многих университетах, научных институтах и технологических компаниях по всему миру начиная с середины 1990-х годов. Известны работы, проводимые как независимыми лабораториями (MIT, Stanford), так и крупными IT-компаниями (Microsoft, IBM, Google), в этом направлении.

Современные технологии играют значительную роль в развитии методов распознавания человеческих эмоций [7], что имеет важное значение в таких областях, как медицина [11], безопасность, маркетинг и интерактивные услуги [5]. Особое внимание уделяется распознаванию эмоций лица [13], задаче, которая включает идентификацию различных эмоциональных состояний, таких как радость, грусть, злость и удивление [4]. В этом контексте методы глубокого обучения, в том числе архитектура AlexNet [8], демонстрируют значительные успехи, позволяя эффективно классифицировать и распознавать эмоциональные выражения.

В данной работе осуществляется обзор и анализ различных открытых решений в области распознавания эмоций [9]. Проводится сравнение их эффективности на таких наборах данных, как AffectNet и Emotional Detection [3], что позволяет оценить их применимость в реальных условиях [6]. Акцент делается на возможностях этих систем адекватно идентифицировать человеческие эмоции [10], что является ключевым аспектом для широкого спектра приложений.

Целью данного исследования является оценка современных методов распознавания эмоций с использованием глубокого обучения [12], а также анализ потенциала и ограничений этих подходов в контексте их применения в различных сферах [7]. Это исследование способствует пониманию текущего состояния технологий распознавания эмоций и выявляет направления для дальнейших улучшений и разработок в этой области [9].

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались некоторые открытые наборы данных. Рассмотрим используемые открытые наборы.

A. Emotional Detection

Обширный набор данных для обучения и тестирования систем распознавания эмоций, включающий 35,685 примеров 48x48 пиксельных черно-белых изображений лиц, представлен в виде разделенных тренировочных и тестовых наборов данных. Изображения категорированы в зависимости от эмоций, выражаемых на лицах (счастье, нейтралитет, грусть, злость, удивление, отвращение, страх) [14].

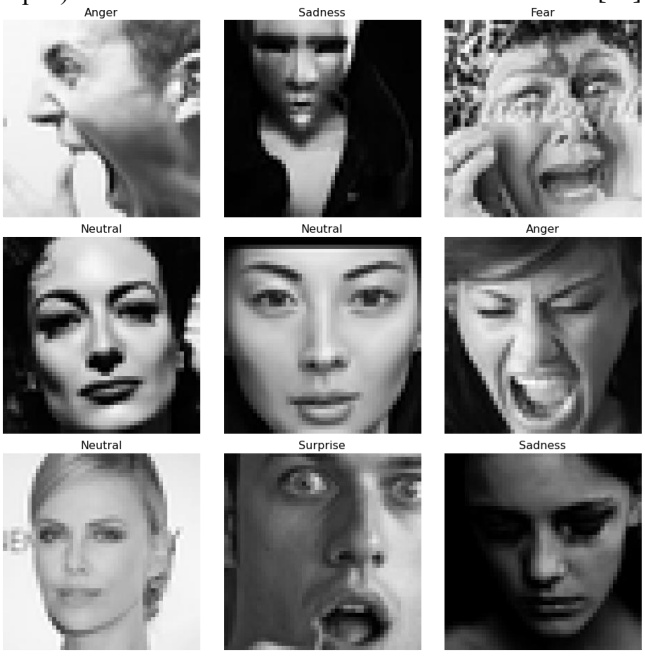


Рис. 1. Примеры классов состояния человека: а) anger, б) Sadness, в) Fear, г) Neutral, д) Surprise, е) Sadness

На рисунке 1 представлены некоторые из этих изображений, демонстрирующие разнообразие выражений лиц и эмоций:

- изменения мимики и тонких нюансов выражений лица, вызванные внутренними чувствами или внешними стимулами (а, б, в, г);

- нестандартные или уникальные выражения лиц, которые могут возникать редко или в особых ситуациях (и);

B. AffectNet

Набор данных AffectNet [12] включает около 1 миллиона размеченных изображений лиц, собранных из Интернета, которые были аннотированы с указанием различных эмоций. Изображения были собраны из разных культур и стран, обеспечивая разнообразие выражений лиц, поз, углов съемки и освещения. Аннотации к данным содержат информацию о:

- очертаниях лиц в ограничивающих прямоугольниках;
- эмоциональном состоянии (тип выраженной эмоции);
- интенсивности эмоций (от слабо выраженной до сильно выраженной);
- демографических характеристиках (пол, возраст);
- наличии визуальных искажений или препятствий (например, очки, маски);
- аспектах, связанных с позой головы и взглядом.

Сбор данных осуществлялся с использованием разнообразных источников, чтобы обеспечить широкое покрытие различных эмоций и условий съемки. Для создания более точных и устойчивых моделей распознавания эмоций, в набор данных включены также дополнительные метаданные, такие как разметка по частям лица и ключевым точкам. Это позволяет исследователям использовать набор данных для разработки и улучшения алгоритмов компьютерного зрения и машинного обучения, направленных на более точное и надежное распознавание эмоций по изображениям лиц.

III. Архитектуры

A. DenseNet169 Transfer Learning

- В работе решается задача распознавания эмоций человека для применения в психологии, маркетинге и интерактивных технологиях. Авторы предлагают использовать нейронную сеть DenseNet169 с механизмом Transfer Learning, обученную на наборе данных AffectNet, для локализации и классификации эмоций на изображениях лиц. В сочетании с предварительно обученными моделями, в которых содержится информация о различных эмоциональных состояниях, система способна предоставлять точные данные о типе и интенсивности эмоции на лице человека.

Разработка системы проходила в две стадии:

- в течение оффлайн-фазы были использованы данные из AffectNet для обучения модели с учетом различных эмоций;
- онлайн-фаза заключается в применении модели для классификации эмоций на новых изображениях, обеспечивая распознавание и интерпретацию эмоциональных состояний (рисунок 2) и их интенсивности – от слабо выраженных до очень ярких.

Важной частью работы является нейросетевая архитектура DenseNet169 (рисунок 3), предоставляющая эффективную структуру для Transfer Learning и классификации эмоций. Модель обучена на 1500 батчах по 64 изображения каждый и настроена на распознавание различных эмоций, таких как радость, грусть, удивление, злость и др. Набор данных AffectNet был использован для обучения и тестирования нейронной сети, обеспечивая высокую точность и обобщающую способность модели.

В процессе обучения применялись техники аугментации изображений и батч-нормализация для улучшения обобщающей способности и устойчивости модели к различным условиям освещения и поз лица. Каждые 10 батчей менялось разрешение изображений для адаптации модели к разным масштабам и углам обзора. Это позволило создать систему, которая точно распознает и классифицирует эмоциональные состояния людей в разнообразных ситуациях и условиях.

Рис. 2. Распределение датасета Emotional Detection

B. AlexNet model

Другой подход заключается в использовании модификации AlexNet для локализации лиц (нахождение их ограничивающих прямоугольников) и последующего анализа выражений лица для распознавания эмоций. Исходный код и модели доступны в рамках исследовательских работ по распознаванию эмоций.

AlexNet – это архитектура нейронной сети, которая была разработана для задач классификации изображений и последующего выявления важных признаков в данных. AlexNet, описанная в работах по глубокому обучению, использует слои свертки для извлечения важных признаков из изображений.

При использовании AlexNet для распознавания эмоций, модель адаптируется для идентификации особенностей лица, связанных с выражением эмоций. Сеть обучается на базе данных AffectNet и других наборах данных по распознаванию эмоций, таких как Emotional Detection, где изображения лиц аннотированы с точки зрения выражаемых эмоций. Это позволяет модели точно локализовать лица на изображении и определять, какие эмоции выражаются, используя для этого информацию об ограничивающих прямоугольниках и выражениях лиц.

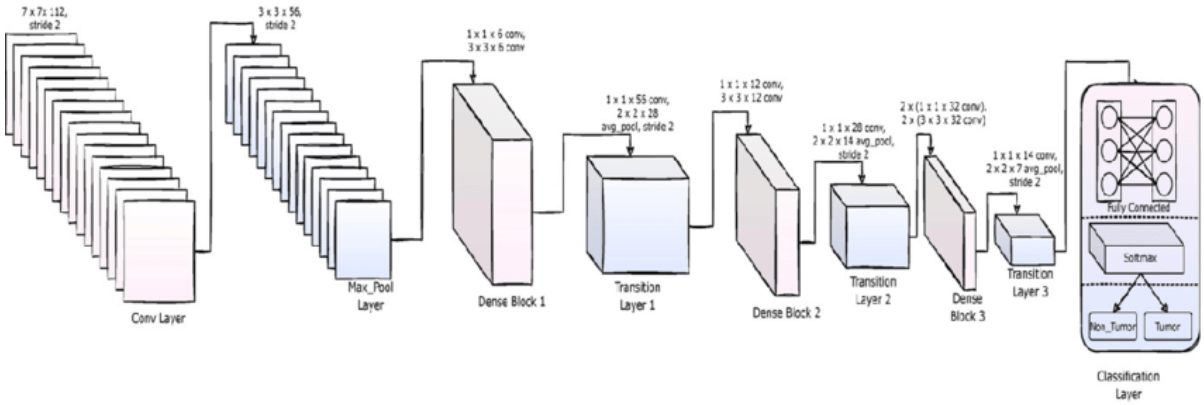


Рис. 3. Архитектура DenseNet169

Итоговая архитектура для задачи распознавания эмоций использует версию AlexNet, представленную на рисунке 4. Модель инициализируется с весами, предобученными на наборе данных ImageNet [6], широко используемом в области компьютерного зрения.

Для локализации и последующего распознавания эмоций на лице применяется та же модель AlexNet.

Для классификации эмоций используются предобученные веса, а модель дообучается на изображениях лиц из наборов данных AffectNet и Emotional Detection, содержащих аннотации различных эмоций. Как и в исходном случае, используется функция потерь кросс-энтропии в мультиклассовой классификации с softmax активацией. Процесс дообучения обычно длится 10 эпох с размером батча в 32 изображения. Выходом данной модели является категория, соответствующая одной из эмоций, аннотированных в используемых наборах данных.

IV. Сравнение

Для сравнения двух подходов в задаче распознавания эмоций использовалась часть наборов данных AffectNet и Emotional Detection – 7473 изображения с размеченными эмоциями. Качество работы подходов оценивается по качеству локализующей и классифицирующей частей модели. Оценка локализации лиц на изображениях проводится с помощью меры Жаккара (Intersection over Union, IoU) для каждой детекции лица. Введены следующие метрики:

- TP (True Positive) – модель, верно, локализовала лицо;
- FP (False Positive) – модель нашла лицо там, где его нет;
- FN (False Negative) – модель не нашла лицо, хотя оно есть.
- Не учитывая TN (True Negative), так как отсутствие лица не является целью задачи, используются следующие функции оценок:
- Precision – доля верно определенных лиц от всех определенных лиц;
- Recall – доля найденных лиц от всех лиц в данных;
- F1-Score – баланс между Precision и Recall.

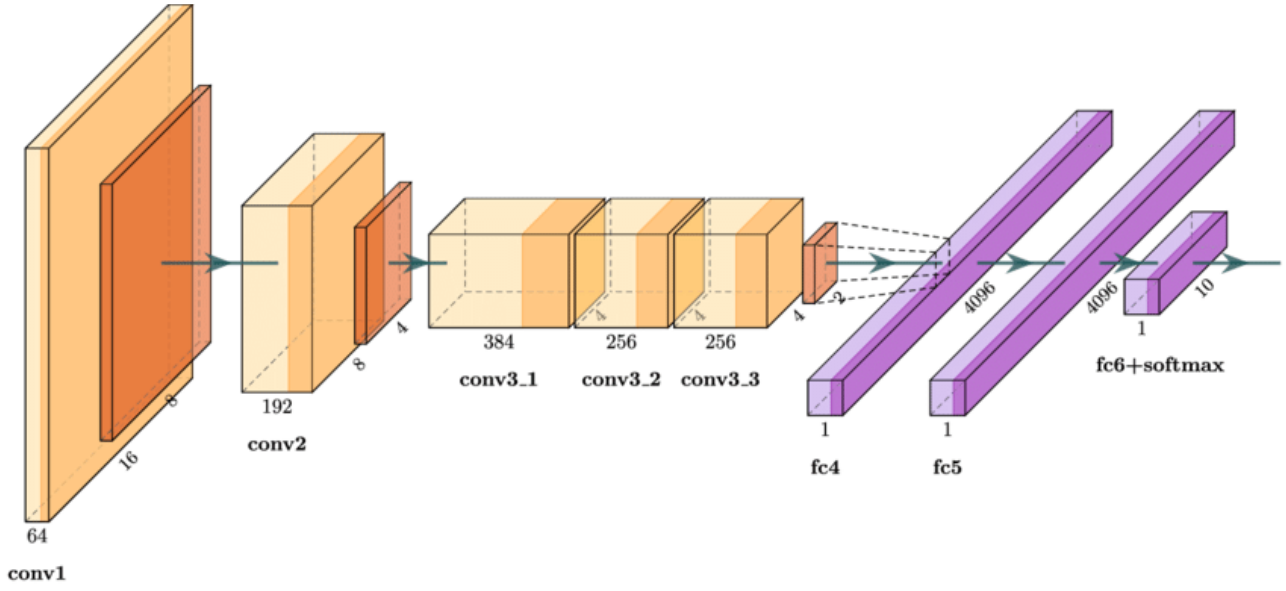


Рис. 4. Архитектура AlexNet

В случае с видеопоследовательностями можно использовать функции MOT, такие как MOTA (multiple object tracking accuracy) для оценки точности отслеживания лиц и MOTP (multiple object tracking precision) для оценки точности локализации лиц. Формулы (2) и (3) соответствуют данным функциям. Из-за специфики задачи, значения функции MOTA могут быть отрицательными, обозначая диапазон значений.

$$MOTA = 1 - \frac{FN+FP+IDS}{GT} \quad (2)$$

$$MOTP = \frac{1}{TP}\sum_i IoU_i \quad (3)$$

Таблица I показывает количественные оценки для двух подходов с использованием модели AlexNet для локализации и классификации эмоций. В оценку классифицирующей части включены все объекты из множества TP локализующей части. Классифицирующая часть модели на выходе имеет различное количество классов в зависимости от того, какие эмоции были определены и размечены в используемых наборах данных.

ТАБЛИЦА I. Оценка детектирующей части

	AlexNet	DenseNet169
TP	19322	3367
FP	2625	10213
FN	23257	11624
Precision	0.88	0.25
Recall	0.86	0.22
F1	0.87	0.24
MOTA	0.74	-0.46
MOTP	0.59	0.44

Из таблицы видно, что детектор на основе модифицированной архитектуры AlexNet показывает значительно более высокие показатели, что говорит о его способности эффективно находить и правильно классифицировать эмоции на изображениях лиц. Модель AlexNet значительно чаще определяет верные эмоциональные состояния и реже ошибается, что делает её более предпочтительной для задач распознавания эмоций. Значение MOTA для AlexNet достаточно высоко по сравнению с упрощенной версией, подтверждая её эффективность.

Классификация эмоций включает в себя все объекты из множества TP, определенного в локализующей части. Модель AlexNet на выходе предсказывает различные классы эмоций. В отличие от классификатора упрощенной AlexNet, которая может распознавать меньше типов эмоций, модифицированная AlexNet обеспечивает более широкий спектр распознаваемых эмоциональных состояний. Матрицы ошибок для обеих моделей имеют разные размеры в зависимости от количества распознаваемых классов эмоций.

На рисунке 5 отчет о классификации для модели AlexNet, включая precision, recall и F1-меру для различных классов эмоций. Номера классов 0–6 могут соответствовать, например, радости, грусти, удивлению, злости, отвращению, страху и нейтральному состоянию соответственно. Это демонстрирует способность модели точно и эффективно классифицировать эмоции на основе изображений лиц, что делает её полезным инструментом в различных приложениях, связанных с анализом человеческого поведения и интерактивных систем.

	precision	recall	f1-score	support
0	0.52	0.61	0.56	958
1	0.00	0.00	0.00	111
2	0.47	0.34	0.39	1024
3	0.87	0.86	0.87	1774
4	0.55	0.69	0.61	1233
5	0.51	0.48	0.49	1247
6	0.75	0.75	0.75	831
accuracy			0.63	7178
macro avg	0.52	0.53	0.52	7178
weighted avg	0.62	0.63	0.62	7178

Рис. 5. Классификация отчёта

Как видно, эмоции радость и грусть распознаются с не слишком высоким значением F1-меры, тогда как другие эмоции практически не распознаются. Это можно объяснить тем, что для точного распознавания большого количества различных эмоций крайне важно иметь обширную и сбалансированную обучающую выборку. Однако такой выборки, возможно, не было при обучении модели AlexNet.

Численные оценки классификации этой нейросети имеют значения, близкие к 100%. Такие высокие показатели могут быть объяснены несколькими факторами: более продолжительным обучением на более релевантных наборах данных, меньшим количеством предсказываемых классов, более широкой и сбалансированной обучающей выборкой, а также большим числом тестовых изображений, так как более качественный детектор смог правильно локализовать и классифицировать больше лиц с эмоциями.

Сравнивая классификаторы на основе AlexNet и других подходов, можно заметить, что наиболее распространенные эмоции, такие как радость и грусть, которые широко представлены в датасете AffectNet, распознаются лучше всего. Это еще раз подчеркивает важность наличия репрезентативной, обширной и сбалансированной обучающей выборки для достижения высокой точности и качества распознавания различных эмоций.

V. Заключение

Основные наборы данных, такие как AffectNet и Emotional Detection, использовались для обучения и тестирования рассматриваемых нейронных сетей, направленных на распознавание эмоций. Были исследованы два подхода к детекции — локализации и классификации эмоций: модифицированная AlexNet, адаптированная авторами для определения различных эмоций, и сочетание различных предобученных моделей для сравнения. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения и использованных наборов данных для обучения и тестирования.

Приведенные подходы были сравнены на выборке из наборов данных AffectNet и Emotional Detection. Отдельно были оценены качество локализации лиц и классификации выраженных эмоций. Исходя из полученных данных, очевидно, что модифицированная AlexNet, адаптированная и обученная авторами, демонстрирует преимущество перед другими подходами. Это объясняется различиями в обучающих процессах и качестве данных, используемых для обучения. Однако следует отметить, что для общего сравнения архитектур необходимы фиксированные наборы данных и унифицированные процессы обучения и тестирования.

Литература

[1] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-CostNavigation System for UAV," 2023 30th Saint Petersburg InternationalConference on Integrated Navigation Systems (ICINS), SaintPetersburg, Russian Federation, 2023, pp. 1-6, doi:10.23919/ICINS51816.2023.10168469.

[2] Ali, B., Sadekov, R.N., Tsodokova, V.V., A Review of NavigationAlgorithms for Unmanned Aerial Vehicles Based on Computer VisionSystems, Gyroscopy and NavigationЭта ссылка отключена., 2022

[3] Guzhva, N.S., Prun, V.E., Postnikov, V.V., Sadekov, R.N., Sholomov,D.L., Using 3D Object Detection DNN in an Autonomous Tram toPredict the Behaviour of Vehicles in the Road Scene, 29th SaintPetersburg International Conference on Integrated Navigation Systems,ICINS 2022

[4] Guzhva, N.S., Ali, B., Bakulev, K.S., Sadekov, R.N., Sholokhov, A.V.Evaluating the Accuracy of Tram Positioning System in High-RiseBuilding Environment Using Data from Visual GeoinformationSystems, 30th Anniversary Saint Petersburg International Conferenceon Integrated Navigation Systems, ICINS 2023, 2023

[5] Li, S. and Deng, W. (2016). "Deep Facial Expression Recognition: A Survey," in IEEE Transactions on Affective Computing, vol. 13, no. 3, pp. 1190–1209.

[6] Kahou, S. E., Pal, C., Bouthillier, X., Froumenty, P., Gülçehre, Ç., Memisevic, R., Vincent, P., Courville, A., Bengio, Y., Ferrari, R. C., Mirza, M., Jean, S., Carrier, P. L., Dauphin, Y., Boulanger-Lewandowski, N., Aggarwal, A., Zumer, J., Lin, Z., Pereira, E., Dupont, S., de Souza, J. R., Cohen, J. P., Côté, M., and Bengio, S. (2015). "Combining Modality Specific Deep Neural Networks for Emotion Recognition in Video," in Proceedings of the 15th ACM on International Conference on Multimodal Interaction, pp. 543–550.

[7] Mollahosseini, A., Hasani, B., and Mahoor, M. H. (2017). "AffectNet: A Database for Facial Expression, Valence, and Arousal Computing in the Wild," IEEE Transactions on Affective Computing, vol. 10, no. 1, pp. 18–31.

[8] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "VisualLocalization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg InternationalConference on Integrated Navigation Systems (ICINS), St. Petersburg,Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[9] Kaya, H., Salah, A. A., (2015). "Deep Learning and Face Recognition: The State of the Art," in Advances in Deep Learning, pp. 1–35.

[10] Lucey, P., Cohn, J. F., Kanade, T., Saragih, J., Ambadar, Z., and Matthews, I. (2010). "The Extended Cohn-Kanade Dataset (CK+): A complete dataset for action unit and emotion-specified expression," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, pp. 94–101.

[11] Goodfellow, I. J., Erhan, D., Carrier, P. L., Courville, A., Mirza, M., Hamner, B., Cukierski, W., Tang, Y., Thaler, D., Lee, D-H., Zhou, Y., Ramaiah, C., Feng, F., Li, R., Wang, X., Athanasakis, D., Shawe-Taylor, J., Milakov, N., Park, J., Ionescu, R., Popescu, M., Grozea, C., Bergstra, J., Xie, J., Romaszko, L., Xu, B., Chuang, Z. J., Bengio, Y. (2013). "Challenges in Representation Learning: A Report on Three Machine Learning Contests," in Neural Networks, vol. 64, pp. 59–63.

[12] Corneanu, C., Simon, M., Cohn, J. F., and Guerrero, S. E. (2016). "Survey on RGB, 3D, Thermal, and Multispectral Approaches for Facial Expression Recognition: History, Trends, and Affect-related Applications," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 38, no. 8, pp. 1548–1568.

[13] Zeng, Z., Pantic, M., Roisman, G. I., and Huang, T. S. (2009). "A Survey of Affect Recognition Methods: Audio, Visual, and Spontaneous Expressions," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 31, no. 1, pp. 39–58.

[14] Ekman, P., (1992). "An argument for basic emotions," Cognition and Emotion, vol. 6, no. 3/4, pp. 169–200.

Вопросы построения карты глубины на основе моно и видеопоследовательности

Д.А. Подгорный
Кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2314956@edu.misis.ru

И. А.Селезенёв
Кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m1902948@edu.misis.ru

Аннотация — данная работа посвящена обзору современных достижений в развитии оценки глубины изображения на основе монопоследовательности. В её рамках рассматриваются популярные методы оценки глубины изображения, изучается устройство каждого подхода, производится оценка результатов работы и сравнение моделей друг с другом.

Ключевые слова — глубина изображения, монопоследовательность, относительная глубина, метрическая глубина, MiDaS, ZoeDepth, стереокинематограф)

I. Введение

Технологии машинного обучения стремительно развиваются[1], ровно как и методы оценки глубины изображения, используемые в них. Каждый год выпускаются новые и новые научные материалы с предложениями по улучшению существующих решений. В рамках данной статьи будет рассмотрена оценка глубины изображения на основе монопоследовательности (Single Image Depth Estimation - SIDE), изучены современные методов, позволяющие сделать оценку глубины более качественной. Для примера будут взяты две научные статьи, выпущенных последовательно - [2] и [3], причём в первой статье предлагались идеи для усовершенствования методов современников, а во второй были предложены возможные методы развития концепции на основе модели из работы [2]. Имплементации методов из рассматриваемых статей называются MiDaS и ZoeDepth соответственно. Указанные модели будут описаны по отдельности, с рассмотрением основных принципов, за счёт которых удалось улучшить оценку глубины - в т.ч. архитектуру модели, применяемые наборы данных, функции потерь, и т.д.

Предметом обзора выбраны две популярные модели - MiDaS и ZoeDepth, а также сопутствующие им научные статьи. MiDaS был предложен сообществу как демонстрация практической осуществимости улучшений в современных моделях оценки глубины за счёт смешения различных наборов данных. В том числе, чтобы улучшить результативность, был составлен набор данных на основе картин стереокинематографа. ZoeDepth, в свою очередь, - это модель, предложенная для оценки глубины изображения с помощью смешения относительных и метрических методов оценки глубины изображения - предшествующие работы фокусируются либо на общей производительности с относительной оценкой глубины, либо на специфических наборах данных (т.е. метрической оценке глубины). Авторы описывают первый подход, комбинирующий два вышеописанных, тем самым предлагая универсальную модель с сохранением метрической системы.

В конце работы будет приведено сравнение индивидуальных результатов каждой модели, а затем их сравнение между собой..

II. Обзор MiDaS

В работе [2] исследуются методы обучения устойчивых моделей монокулярной оценки глубины, предназначенных для работы в разнообразных условиях. Описывается разработка новой функции потерь, инвариантной к основным источникам несовместимости между наборами данных, включая неизвестные и несогласованные масштабы и базовые показатели. Представленная система расчета потерь делает возможным обучение на данных, полученных с использованием различных сенсорных модальностей, таких как стереокамеры (с возможно неизвестной калибровкой), лазерные сканеры и сенсоры структурированного света.

Основной смысл работы состоит в описании смешения наборов данных из разных источников для получения наибольшей эффективности и универсальности модели для оценки глубины изображения как в закрытых помещениях, так и в открытых пространствах. Обучение происходило на разнообразных наборах данных из открытых источников, но также авторы предложили свой, основанный на 3Dфильмах, состоящий из 21 произведения кинематографа.

Были предложены различные наборы данных, которые подходят для монокулярной оценки глубины, т.е. они состоят из изображений RGB с соответствующей аннотацией глубины той или иной формы. Наборы данных различаются по захваченным средам и объектам (внутренние/наружные сцены, динамические объекты), по типу аннотации глубины (разреженная/плотная, абсолютная/относительная глубина), по точности (лазер, время полета, Fm, стерео, аннотация человека, синтетические данные), качеству изображения и настройкам камеры, а также размер набора данных.

A. Существующие проблемы

Получить точные данные в больших масштабах проблематично, и ещё сложнее получить такие данные о движущихся объектах. При этом, даже одного качественного набора данных будет недостаточно, поскольку, как показывает практика, модель, обученная на единственном наборе данных, будет показывать хорошие результаты на подмножестве, отделённого от обучающего набора, но будет проваливаться в эффективности своей работы на примерах извне. Это связано с тем, что каждый набор данных отличается от других обширным списком параметров, которые

становятся чувствительными для модели на практике. Так, например, на процесс могут повлиять модель камеры, характеристики объектива и характеристики другого оборудования, которое может использоваться при оценке истинной глубины. Для решения данной проблемы авторы внимательно выбирали обучающие наборы данных. Проводились эксперименты с пятью дополняющими друг друга наборами данных для обучения - они смешивались либо поровну, либо по Парето-оптимальному принципу:

1. ReDWeb [4] - небольшой набор тщательно отобранных статичных и динамичных сцен с истинной глубиной, полученной из стереопоследовательности;
2. MegaDepth [5] - значительно больший набор, но преимущественно состоящий из статичных объектов, однако с качественной оценкой глубины, полученной за счёт измерения информации с большого количества различных ракурсов;
3. WSVDMegaDepth [6] состоит из стереофонических видеороликов, полученных из Интернета, и содержит разнообразные динамичные сцены;
4. DIML Indoor [7] - это набор RGB-D изображений, преимущественно статичных сцен в помещении, захваченных с помощью Kinect v2;
5. 3D Movies - набор из различных произведений стереокинематографа; это новый набор данных, предложенный командой разработчиков MiDaS, о нём - далее.

B. Набор на основе стереокинематографа

С целью дополнить существующий комплект наборов данных, авторы решили составить набор из трёхмерного кино; изначально фильмы не предлагают готовую метрическую оценку глубины, по этой причине авторы вычисляют относительную глубину на основе карты несоответствий, полученной из стереопоследовательности.

Во время разработки набора данных авторам пришлось столкнуться с некоторыми испытаниями. Так, например, диапазон несоответствий может разниться даже внутри одного произведения - зачастую, в стереокино несоответствие значительно возрастает в начале и конце картины для того, чтобы на короткое время вызвать очень заметный стереоскопический эффект. В это же время середина произведения - напротив, снижает диапазон несоответствий, чтобы обеспечить комфортный просмотр зрителю.

Как следствие, фокусные расстояния, ракурсы записи и угол схождения между камерами стереоустановки неизвестны и варьируются в зависимости от сцены даже в пределах одного фильма. Помимо этого, глубина может зависеть от характеристик комплекта для стереозаписи, а также от стилистических решений (например, когда монтажёр смещает пары изображений относительно друг друга).

Чтобы облегчить эти проблемы, авторы применяют современный алгоритм оптического потока к стереопарам. Он способен обрабатывать положительные и отрицательные несоответствия, которыми обладают записи, полученные с помощью стереоустановок для видеосъёмки. Также сцены обрезаются под унифицированный размер 1880x800 пикселей. При этом для набора данных выбираются фильмы с натуральной стереосъёмкой (фильмы, снятые на одиночную камеру с постпродакшном под 3D-кинотеатры отсеиваются) и только в blu-ray формате для получения качественных кадров.

Кадры выбираются с установкой собрать качественный и разнообразный набор данных - при этом отбрасываются кадры из хаотичных динамических сцен, и диалогов.

Далее было необходимо определить loss-функцию, достаточно гибкую и способную работать в следующих обстоятельствах:

- Различные оценки глубины - прямые и обратные.
- Различие масштабов.
- Разнообразие смещений.

Авторы предложили выполнить прогнозирование в пространстве несоответствий (обратная глубина вплоть до масштаба и сдвига) вместе с семейством loss-функций, не зависящих от сдвига и масштаба, чтобы справиться с вышеупомянутыми трудностями.

Пусть M обозначает количество пикселей в изображении с допустимой базовой истинностью, и пусть Θ - параметры модели прогнозирования. Пусть d = d(Θ) ∈ RM - прогноз несоответствия, и пусть d∗ ∈ RM – соответствующее несоответствие базовой истинности. Отдельные пиксели индексируются по нижним индексам. Тогда, потери, не зависящие от масштаба и сдвига, для одного образца как:

$$\mathcal{L}_{ssi}(\hat{d}, \hat{d}^*) = \frac{1}{2M} \sum_{i=1}^{M} \rho\left(\hat{d}_i - \hat{d}_i^*\right)$$

где d и d ∗ - масштабированные и сдвинутые версии предсказаний, и эталонные данные, а ρ определяет конкретный тип функции потерь.

Пусть s : RM → R+ и t : RM → R обозначают оценки масштаба и смещения. Чтобы определить значимые потери, важным требованием является то, что предсказание и базовая достоверность должны быть соответствующим образом согласованы относительно их масштаба и сдвига, т.е. нам нужно гарантировать, что s(ˆd) ≈ s(ˆd ∗) и t(ˆd) ≈ t(ˆd ∗).

Авторы предлагают две разные стратегии для выполнения этого выравнивания. Первый подход приводит прогноз в соответствие с эталонными данными на основе критерия наименьших квадратов:

$$(s,t) = \arg\min_{s,t} \sum_{i=1}^{M} (sd_i + t - d_i^*)^2,$$

$$\hat{d} = sd + t,\ \hat{d}^* = d^*,$$

где ˆd и ˆd ∗ - выровненное предсказание и эталонные данные соответственно.

Факторы s и t могут быть эффективно определены, если записать смещённые и масштабированные оценки глубины в форме: ⃗di = (di , 1)⊤ и h = (s, t) ⊤ и решить:

$$h^{opt} = arg\min_{s,t} \sum_{i=1}^{M} (\vec{d}_i^{\top} h - d_i^*)^2$$

как задачу наименьших квадратов, что можно записать как:

$$h^{opt} = \left(\sum_{i=1}^{M} \vec{d}_i \vec{d}_i^{\top} \right)^{-1} \left(\sum_{i=1}^{M} \vec{d}_i \vec{d}_i^* \right).$$

Авторы установили ρ(x) = ρmse(x) = x2, чтобы определить инвариантный к масштабу и смещению наименьший квадрат отклонения. Данную loss-функцию обозначили как Lssimse.

Поскольку наименьший квадрат отклонения очень чувствителен к резко отличающимся от общей массы значениям, в работе предложили ввести более надёжную loss-функцию для улучшения обучения. Для этого оценка смещения вычисляется как медианное значение, а оценка масштаба как среднее арифметическое модуля разности глубины и смещения:

$$t(d) = median(d),\ s(d) = \frac{1}{M} \sum_{i=1}^{M} |d - t(d)|.$$

Оценка и истина нормируются для корректной обработки в дальнейшем:

$$\hat{d} = \frac{d - t(d)}{s(d)},\ \hat{d}^* = \frac{d^* - t(d^*)}{s(d^*)}.$$

Определяются две надёжные loss-функции. Первая, Lssimrim, вычисляет модуль отклонений ρmae(x) = |x|. Вторая, Lssitrim, образуется путём отбрасывания наибольших 20% отклонений в каждом изображении, независимо от их величины, что вычисляется как:

$$\mathcal{L}_{ssitrim}(\hat{d}, \hat{d}^*) = \frac{1}{2M} \sum_{j=1}^{U_m} \rho_{mae} \left(\hat{d}_j - \hat{d}_j^* \right),$$

где | ˆdj − ˆd ∗ j | ≤ | ˆdj+1− ˆd ∗ j+1|, a Um = 0.8M. Таким образом, слишком большие отклонения не будут влиять на процесс обучения.

Чтобы стимулировать более плавные изменения градиента и более резкие разрывы глубины на прогнозируемой карте глубин (см. рис. 1), авторы вводят многомасштабную масштабно-инвариантную функцию для сопоставления градиента [8] Lreg, определяемую, как штраф в размере l за различия в градиентах логарифмической глубины между прогнозируемой и истинной картой глубины:

$$\mathcal{L}_{reg}(\hat{d}, \hat{d}^*) = \frac{1}{M} \sum_{k=1}^{K} \sum_{i=1}^{M} (|\nabla_x R_i^k| + |\nabla_y R_i^k|),$$

где Ri = | ˆd − ˆd ∗ |, и Rk обозначает различие в картах несоответствий в масштабе k. Авторы используют K = 4 уровня масштабирования, уменьшая разрешение изображения вдвое на каждом уровне.

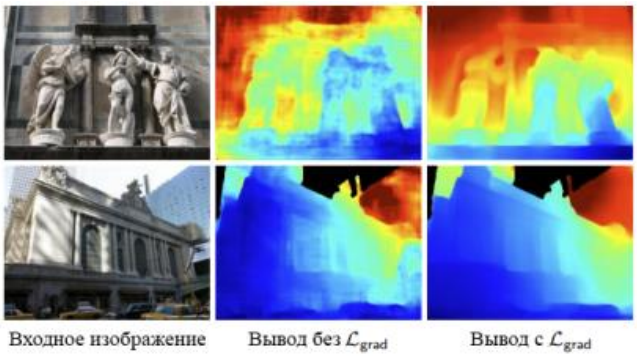


Рис. 1 Эффект от применения функции Lgrad (идентичной и Lreg) из статьи, в которой данная функция была представлена изначально.

Так, итоговая loss-функция для набора l выглядит следующим образом:

$$\mathcal{L}_l = \frac{1}{N_l} \sum_{n=1}^{N_l} \mathcal{L}_{ssi}(\hat{d}^n, (\hat{d}^*)^n) + \alpha\, \mathcal{L}_{reg}(\hat{d}^n, (\hat{d}^*)^n),$$

где Nl - размер обучающего набора данных и α = 0.5.

III. Обзор ZoeDepth

Работа разделена на две части - метрическая оценка глубины (MDE) и относительная оценка глубины (RDE). MDE имеет широкое прикладное применение, однако производительность может резко падать, когда модель обучается на наборах данных с большим разбросом в метрическом диапазоне (например, наборы с изображениями помещений и наоборот, снятые снаружи). По этой причине, MDE специфичны к определённому набору данных и плохо себя показывают на разнообразных массивах данных. В RDE решена проблема привязанности к определённому набору данных, но, в то же время, оценочная глубина не имеет метрики, что накладывает соответствующие прикладные ограничения на применение данных моделей.

Авторы предлагают двухэтапный алгоритм, объединяющий MDE и RDE. На первом этапе модель проходит дообучение на широком спектре наборов данных, что даёт большую универсальность. Во втором этапе вводятся заголовочные модули (heads) - каждый из которых отведён под отдельный диапазон метрик. Во время инференса встроенный классификатор перенаправляет вывод на соответствующий заголовочный модуль, который возвращает метрическую оценку глубины. Особенность предложенного метода заключается в том, что вывод состоит не из набора одиночных значений глубины на пиксель, а из целого множества значений глубины на пиксель (авторы предлагают называть каждое такое множество "bin которое далее будет переводится как "корзина"). Это позволяет эффективно оценить метрику для относительной глубины изображения.

Предложенный фреймворк может быть представлен в различных конфигурациях. Авторы выделяют три наиболее заслуживающих внимания:

1. Метрическая оценка глубины по одиночному изображению.

2. Метрическая оценка глубины по одиночному изображению с дообучением для оценки относительной глубины.
3. Универсальная метрическая оценка глубины по одиночному изображению с автоматической классификацией.

A. *Метрическая оценка глубины по одиночному изобра жению*

The Модель ZoeD − X − N на основе набора данных NYU Depth v2 [9] без использования относительного предобучения, но с использованием представленных "корзин который одним этим дополнением уже демонстрирует превосходство над современной сетью NeWCRFs [10] на 13.7% в задачах оценки глубины изображений на открытой местности.

B. *Метрическая оценка глубины по одиночному изображению с дообучением для оценки относительной глубины.*

Проведя дообучение для определения относительной глубины на 12 наборах данных, с последующим дообучением для определения метрики на наборе NYU Depth v2, разработчики получили модель ZoeD-M12-N, которая превосходит предыдущую модель на 8.5%, что результирует в превосходство на 21.2% относительно NeWCRFs.

C. *Универсальная метрическая оценка глубины по одиночному изображению с автоматической классификацией*

Модель, аналогичная предыдущей, но с дообучением на двух моделях - к обучающему набору NYU Depth v2 добавляется набор KITTI [11]. Внедрение данного набора данных значительно увеличивает результативность модели в работе с изображениями природы. Результирующая модель ZoeD-M12-NK превосходит другие современные модели, также обученные на NYU Depth v2 и KITTI - показатель абсолютной относительной ошибки (REL) уменьшился на 24.3%, а проверка на 7 не виданных прежде наборах данных демонстрирует превосходство над другими современными моделями, вплоть до 976.4% улучшения оценки метрики в частных случаях.

D. *Техническое описание*

Для относительной оценки глубины разработчики использовали MiDaS. Он обладает рядом преимуществ, рассмотренных ранее. Среди них - инвариантность loss-функции к входным данным разных размерностей, а также парето-оптимальность использования нескольких наборов данных. Для базовой части модели используется архитектура кодировщик-декодировщик, реализованная с помощью модифицированной модели DPT [12], в которой была произведена замена кодировщика на более актуальный в лице модели BEIT [13]. Глубина, как было сказано ранее, определяется с помощью адаптированных "корзин"LocalBins - подхода, изначально предложенного в [14]. В конце модель проходит сквозное дообучение.

E. *LocalBins*

Авторы в своей модели демонстрируют модуль "Metric bins который был вдохновлён методом "LocalBins". По этой причине следует в первую очередь описать принцип работы LocalBins, прежде чем перейти к рассмотрению метода, предложенного в обозреваемой сети.

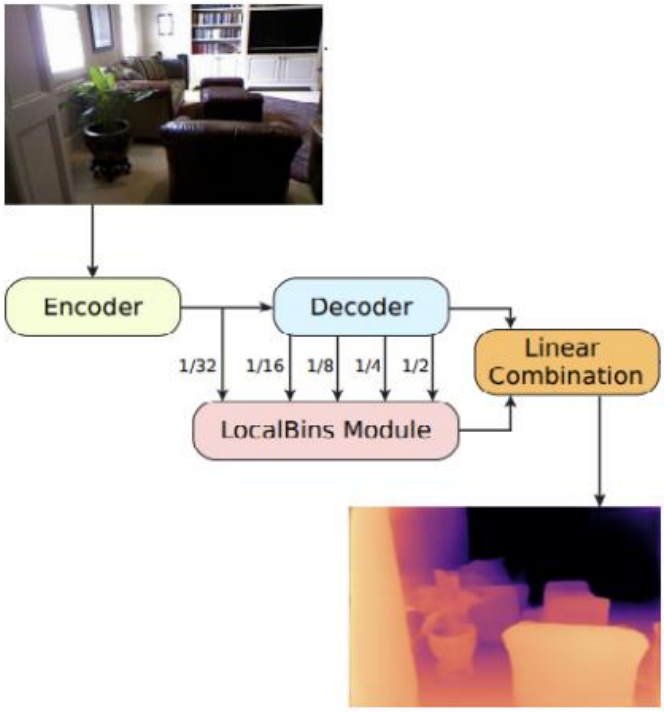


Рис. 2 Архитектура LocalBins

"Корзины" представляют собой диапазон возможных значений глубины пикселя. LocalBins задействует стандартную архитектуру кодировщик-декодировщик на базовом уровне модели (см. рис. 2), и добавляет модуль, использующий многоразмерные функции для предсказания центров корзин для каждого пикселя.

Итоговая глубина вычисляется как линейная комбинация центров, умноженных на собственные вероятностные веса. Эти веса вычисляются с помощью функции softmax, также называемой многопеременной логистической функцией[15].

Изначально модуль определяет стартовое количество "корзин" Nseed на каждый пиксель. Затем, на каждом слое корзины разделяются надвое, что в результате прохода через n слоёв приводит к Ntotal = 2nNseed центрам, которые затем преобразуются в итоговое значение глубины согласно упомянутой линейной комбинации (см. рис. 3).

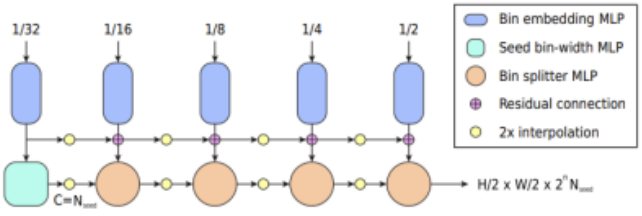


Рис. 3 Принцип работы LocalBins

F. *Metric Bins*

В отличие от LocalBins, авторы предлагают не разделять веса, а настраивать их, двигая "корзину"влево

или вправо на интервале глубины. Модуль, возвращающий метрическую глубину пикселя,

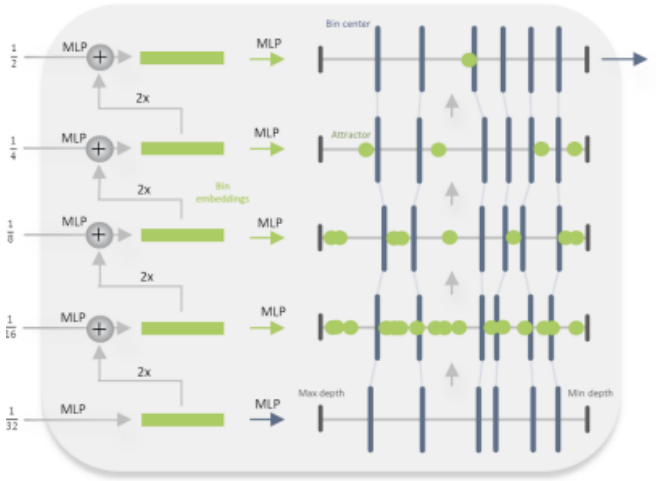


Рис. 4 Принцип работы Metric Bins

принимает на вход 5 каналов, каждый из которых соответствует определённому диапазону иерархии. Значения, приходящие с этих каналов, векторизуются с помощью многослойного перцептрона, а также проходят балансировку и сложение согласно изображению 4.

Эмбеддинг нижнего канала задаёт центральное значение глубины корзины, в то время как остальные эмбеддинги предоставляют т.н. "аттракторы"(о них - далее), изменяющие центр согласно уравнениям, зависящих от весов на каждом аттракторе. Аттракторы формируются с помощью многоразмерных функций, которые прогнозируют точки, к которым притягиваются центры при переходе в другой диапазон иерархии.

Вместо softmax для предсказания распределения вероятностей глубины, авторы решили использовать биноминальное распределение. Это связано с тем, что softmax хорошо работает с неупорядочнными классами, а поскольку "корзины" по своему существу упорядочены, то это может привести к совершенно разным вероятностям у близлежащих пикселей. Для разрешения этой проблемы было принято решение использовать биноминальное распределение, которое затем нормализуется для получения стабилизированных оценок.

В ZoeDepth используется инвариантная к масштабу loss-функция Lpixel, также, как и в LocalBins. Но, в отличие от LocalBins, не используется фасочная loss-функция из-за больших требований к памяти и незначительного улучшения результата.

IV. Результаты и сравнение

A. *Результаты MiDaS*

Авторы MiDas для тестирования предлагают 5 различных вариантов смешения наборов данных (см. рис. 5).

Для того, чтобы иметь общее представление об эффективности каждого набора данных по отдельности, для них были приведены таблицы эффективности в

Mix	RW	DL	MV	MD	WS
MIX 1	✓	✓			
MIX 2	✓	✓	✓		
MIX 3	✓	✓	✓	✓	
MIX 4	✓	✓		✓	✓
MIX 5	✓	✓	✓	✓	✓

Рис. 5 Варианты наборов данных

абсолютных (рис. 6) и относительных (рис. 7) выражениях. Измерения проводились на оценке наборов данных, прежде не встреченных итоговой моделью - DIW [16], ETH3D [17], Sintel [18], KITTI [11], NYU [9] и TUM [19]. Эффективность оценивалась по МНК (Mean).

	DIW	ETH3D	Sintel	KITTI	NYU	TUM	Mean [%]
RW → RW	**14.6**	0.2	**0.3**	28.0	18.7	21.7	—
RW → DL	−37.6	2.0	−4.3	−73.0	**32.3**	**19.4**	−10.2
RW → MV	−26.1	−15.9	−15.5	**10.1**	−10.2	−3.5	−10.2
RW → MD	−31.5	**4.0**	−9.7	−24.3	−1.7	−52.0	−19.2
RW → WS	−32.4	−29.8	−2.9	−34.5	−31.9	3.2	−21.4

Рис. 6 Относительная эффективность одиночных наборов

Изображения 8 и 9 показывают, что, в отличие от использования отдельных наборов данных, смешивание нескольких обучающих наборов последовательно улучшает производительность по отношению к базовому уровню.

	DIW WHDR	ETH3D AbsRel	Sintel AbsRel	KITTI δ>1.25	NYU δ>1.25	TUM δ>1.25
RW → RW	**14.59**	0.151	**0.349**	27.95	18.74	21.69
RW → DL	20.08	0.148	0.364	48.35	**12.68**	**17.48**
RW → MV	18.39	0.175	0.403	**25.12**	20.65	22.44
RW → MD	19.18	**0.145**	0.383	34.73	19.05	32.96
RW → WS	19.31	0.196	0.359	37.59	24.72	20.99

Рис. 7 Абсолютная эффективность одиночных наборов

В то же время добавление наборов данных не приводит к безусловному повышению производительности при использовании интуитивного смешивания.

	DIW	ETH3D	Sintel	KITTI	NYU	TUM	Mean [%]
RW	14.6	0.2	0.3	28.0	18.7	21.7	—
MIX 1	10.9	9.9	−3.7	**18.0**	41.4	33.0	18.3
MIX 2	6.7	8.6	3.2	9.2	40.8	35.7	17.3
MIX 3	**13.5**	10.6	4.9	13.9	**43.8**	29.1	19.3
MIX 4	11.7	11.3	5.2	11.3	38.8	35.5	19.0
MIX 5	12.3	**12.6**	**7.2**	9.1	38.5	**37.2**	**19.5**

Рис. 8 Относительная эффективность смешанных наборов

	DIW WHDR	ETH3D AbsRel	Sintel AbsRel	KITTI δ>1.25	NYU δ>1.25	TUM δ>1.25
RW	14.59	0.151	0.349	27.95	18.74	21.69
MIX 1	13.00	0.136	0.362	**22.91**	10.98	14.53
MIX 2	13.62	0.138	0.338	25.39	11.10	13.94
MIX 3	**12.62**	0.135	0.332	24.06	**10.54**	15.39
MIX 4	12.88	0.134	0.331	24.78	11.46	14.00
MIX 5	12.79	**0.132**	**0.324**	25.41	11.52	**13.62**

Рис. 9 Абсолютная эффективность смешанных наборов

Изображения 10 и 11 демонстрируют результаты Парето-оптимального смешения набора данных для обучения.

	DIW	ETH3D	Sintel	KITTI	NYU	TUM	Mean [%]
RW	14.6	0.2	0.3	28.0	18.7	21.7	—
MIX 1	9.4	7.3	-7.7	13.2	44.1	33.2	16.6
MIX 2	14.1	8.6	0.9	**17.5**	45.5	32.0	19.8
MIX 3	15.8	11.9	5.2	11.7	47.8	32.4	20.8
MIX 4	15.4	13.9	1.7	17.2	43.4	**38.2**	21.6
MIX 5	**15.9**	**14.6**	**6.3**	14.5	**49.0**	34.1	**22.4**

Рис. 10 Относительная эффективность Парето оптимальных наборов данных

	DIW WHDR	ETH3D AbsRel	Sintel AbsRel	KITTI δ>1.25	NYU δ>1.25	TUM δ>1.25
RW	14.59	0.151	0.349	27.95	18.74	21.69
MIX 1	13.22	0.140	0.376	24.26	10.48	14.50
MIX 2	12.54	0.138	0.346	**23.05**	10.21	14.76
MIX 3	12.29	0.133	0.331	24.68	9.78	14.66
MIX 4	12.35	0.130	0.343	23.13	10.61	**13.41**
MIX 5	**12.27**	**0.129**	**0.327**	23.90	**9.55**	14.29

Рис. 11 Абсолютная эффективность Парето-оптимальных

Как можно наблюдать, Парето-оптимальное смешение является более эффективным методом подбора набора данных, при этом из всех вариантов наилучший результат показывает Парето-оптимальное смешение всех пяти наборов данных.

Таким образом, предложенная модель превосходит остальные за счёт смешения наборов обучающих данных.

B. Результаты ZoeDepth

Авторы заявляют, что предложенная ими новая архитектура превосходит самые современные аналоги без использования каких-либо дополнительных данных для дообучения. В качестве подкрепления своих слов они прилагают результаты работы ZoeD-X-N в популярном бенчмарке NYU Depth v2 (см. рис. 12)

Method	$\delta_1\uparrow$	$\delta_2\uparrow$	$\delta_3\uparrow$	REL$\downarrow$	RMSE$\downarrow$	$\log_{10}\downarrow$
Eigen *et al.* [9]	0.769	0.950	0.988	0.158	0.641	–
Laina *et al.* [19]	0.811	0.953	0.988	0.127	0.573	0.055
Hao *et al.* [13]	0.841	0.966	0.991	0.127	0.555	0.053
DORN [11]	0.828	0.965	0.992	0.115	0.509	0.051
SharpNet [31]	0.836	0.966	0.993	0.139	0.502	0.047
Hu *et al.* [14]	0.866	0.975	0.993	0.115	0.530	0.050
Lee *et al.* [22]	0.837	0.971	0.994	0.131	0.538	–
Chen *et al.* [8]	0.878	0.977	0.994	0.111	0.514	0.048
BTS [20]	0.885	0.978	0.994	0.110	0.392	0.047
Yin *et al.* [48]	0.875	0.976	0.994	0.108	0.416	0.048
AdaBins [5]	0.903	0.984	0.997	0.103	0.364	0.044
LocalBins [6]	0.907	0.987	0.998	0.099	0.357	0.042
Jun *et al.* [16]	0.913	0.987	0.998	0.098	0.355	0.042
NeWCRFs [50]	0.922	0.992	0.998	0.095	0.334	0.041
ZoeD-X-N	0.946	0.994	**0.999**	0.082	0.294	0.035
ZoeD-M12-N	**0.955**	**0.995**	**0.999**	**0.075**	**0.270**	**0.032**
ZoeD-M12-NK	0.953	**0.995**	**0.999**	0.077	0.277	0.033

Рис. 12 Результаты бенчмарка NYU Depth v2

Согласно приведённым данным, недообученная модель уже превосходит NeWCRFs на 13.7%. В это же время модель ZoeD-M12-N по тем же данным превосходит NeWCRFs практически на 21%. Помимо более высоких значений, модель работает качественно лучше, выстраивая карту глубины с более чёрткими границами между объектами.

В это же время, модель ZoeD-M12-NK превосходит NeWCRFs на 18.9%, что хуже, чем модель ZoeD-M12-N, однако в противовес данная модель производительнее и универсальнее. Чтобы подчеркнуть сложность обучения способом представленным с ZoeD-M12-NK, авторы решили тем же образом обучить современные аналоги и сравнить результаты (см. рис. 13).

Согласно отчёту, в таких условиях некоторые модели и вовсе отказались запускаться (AdaBins и PixelBins). У других же наблюдалось падение в производительности вплоть до 15%, в то время как ZoeD-M12-NK теряет лишь 8% производительности в сравнении с ZoeD-M12-N. Это демонстрирует большую устойчивость к потере качества при многомерном обучении в сравнении с современниками. Более того, с использованием классификации данный разрыв уменьшается вплоть до 2.6% превосходя NeWCRFs на 25.2% в рамках данного бенчмарка.

Method	NYU	KITTI	iBims-1	vKITTI-2	mRI$_D$
Baselines: no modification					
DORN-X-NK†	0.156	0.115	0.287	0.259	-45.7%
LocalBins-X-NK†	0.245	0.133	0.296	0.265	-74.0%
PixelBins-X-NK†	-	-	-	-	-
NeWCRFs-X-NK†	0.109	0.076	0.189	0.190	0.0%
Baselines: modified to use our pre-trained DPT-BEiT-L as backbone					
DORN-M12-NK†	0.110	0.081	0.242	0.215	-12.2%
LocalBins-M12-NK†	0.086	0.071	0.221	0.121	11.8%
PixelBins-M12-NK†	0.088	0.071	0.232	0.119	10.1%
NeWCRFs-M12-NK†	0.088	0.073	0.233	0.124	8.7%
Ours: different configuations for fair comparison					
ZoeD-X-NK†	0.095	0.074	0.187	0.184	4.9%
ZoeD-M12-NK†	0.081	0.061	0.210	0.112	18.8%
ZoeD-M12-NK	**0.077**	**0.057**	**0.186**	**0.105**	**25.2%**

Рис. 13 Результаты бенчмарка NYU Depth v2 при обучении

Помимо данных результатов, авторы решили провести расчёт универсальности модели путём оценки всех скалярных характеристик различных моделей методом наименьших квадратов. Для этого тестирование проводилось на незнакомых моделям наборах данных - с изображениями закрытых пространств и открытых. В рамках такого сравнения ZoeD-M12-NK показывает небольшое превосходство над современниками, варьируясь в МНК от 5.3% в отношении набора HyperSim до 46.3% в отношении DIODE Indoor (см. рис. 14).

На наборах изображений открытых пространств Virtual KITTI 2 [20] и DIML Outdoor [21] модель NK демонстрирует превосходство над NeWCRFs в размерах от 7.8% до 976.4% соответственно (см. рис. 15). Такое высокое значение в последнем случае объясняется тем, что все модели, кроме NK, были дообучены на наборе KITTI с большими значениями глубины, в то время как DIML Outdoor представляет из себя набор изображений объектов пусть и в открытой местности, но снятых вблизи к камере, что делает их похожими на снимки внутри помещения - поэтому остальные модели показывают плохой результат, в то время как ZoeD-M12-NK демонстрирует свою гибкость.

Method	SUN RGB-D $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	iBims-1 Benchmark $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	DIODE Indoor $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	HyperSim $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$
BTS [20]	0.740	0.172	0.515	-14.2%	0.538	0.231	0.919	-6.9%	0.210	0.418	1.905	2.3%	0.225	0.476	6.404	-8.6%
AdaBins [5]	0.771	0.159	0.476	-7.0%	0.555	0.212	0.901	-2.1%	0.174	0.443	1.963	-7.2%	0.221	0.483	6.546	-10.5%
LocalBins [6]	0.777	0.156	0.470	-5.6%	0.558	0.211	0.880	-0.7%	0.229	0.412	1.853	7.1%	0.234	0.468	6.362	-6.6%
NeWCRFs [50]	0.798	0.151	0.424	0.0%	0.548	0.206	0.861	0.0%	0.187	0.404	1.867	0.0%	0.255	0.442	6.017	0.0%
ZoeD-X-N	0.857	0.124	0.363	13.2%	**0.668**	0.173	0.730	17.7%	**0.400**	**0.324**	**1.581**	**49.7%**	0.284	0.421	5.889	6.1%
ZoeD-M12-N	**0.864**	**0.119**	**0.346**	**16.0%**	0.658	**0.169**	**0.711**	**18.5%**	0.376	0.327	1.588	45.0%	**0.292**	**0.410**	**5.771**	**8.6%**
ZoeD-M12-NK	0.856	0.123	0.356	13.9%	0.615	0.186	0.777	10.6%	0.386	0.331	1.598	46.3%	0.274	0.419	5.830	5.3%

Рис. 14 Результаты тестирования различных моделей на наборах данных, собранных из изображений закрытых пространств.

Method	Virtual KITTI 2 $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	DDAD $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	DIML Outdoor $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$	DIODE Outdoor $\delta_1\uparrow$	REL↓	RMSE↓	$\text{mRI}_\theta\uparrow$
BTS [20]	0.831	0.115	5.368	2.5%	0.805	0.147	7.550	-17.8%	0.016	1.785	5.908	24.3%	0.171	0.837	10.48	-4.8%
AdaBins [5]	0.826	0.122	5.420	0.0%	0.766	0.154	8.560	-26.7%	0.013	1.941	6.272	9.7%	0.161	0.863	10.35	-7.2%
LocalBins [6]	0.810	0.127	5.981	-5.3%	0.777	0.151	8.139	-23.2%	0.016	1.820	6.706	19.5%	0.170	0.821	10.27	-3.6%
NeWCRFs [50]	0.829	0.117	5.691	0.0%	**0.874**	**0.119**	**6.183**	**0.0%**	0.010	1.918	6.283	0.0%	0.176	0.854	9.228	0.0%
ZoeD-X-K	0.837	0.112	5.338	3.8%	0.790	0.137	7.734	-16.6%	0.005	1.756	6.180	-13.3%	0.242	0.799	7.806	19.8%
ZoeD-M12-K	**0.864**	**0.100**	**4.974**	**10.5%**	0.835	0.129	7.108	-9.3%	0.003	1.921	6.978	-27.1%	**0.269**	0.852	**6.898**	**26.1%**
ZoeD-M12-NK	0.850	0.105	5.095	7.8%	0.824	0.138	7.225	-12.8%	**0.292**	**0.641**	**3.610**	**976.4%**	0.208	**0.757**	7.569	15.8%

Рис. 15 Результаты тестирования различных моделей на наборах данных, собранных из изображений открытых пространств.

Таким образом, обучение сразу на нескольких различных множествах - в текущем случае, на изображениях как закрытых помещений, так и открытой местности, показывает свою важность и эффективность.

C. Сравнение MiDaS и ZoeDepth

Headings, or heads, are organizational devices that guide the reader through your paper. There are two types: component heads and text heads.

Для сравнения двух обозреваемых моделей были развёрнуты авторские модели из официальных репозиториев.

Стоит отметить, что авторы MiDaS в рамках своей работы предлагают несколько вариантов моделей. Под MiDaS будет подразумеваться модель DPT-Large, под ZoeDepth - ZoeD-M12-NK.

Честное сравнение по количественным характеристикам провести затруднительно, потому что ZoeDepth, как было сказано, предоставляет метрическую карту глубины, в то время как MiDaS – относительную. По этой причине результирующие значения глубин из двух моделей будут друг от друга отличаться - см. рис. 16. Тем не менее, относительную глубину можно привести к метрической. Для этого был вычислен коэффициент k линейного преобразования глубины из MiDaS в диапазон ZoeDepth как отношение суммы глубин ˆdz ZoeDepth к сумме глубин ˆdm MiDaS:

$$k = \frac{\sum_{i,j} \hat{d_z^{ij}}}{\sum_{i,j} \hat{d_m^{ij}}}$$

После отображения линейного преобразования на матрицу глубины MiDaS, обе карты глубины отображаются на диапазон от 0 до 1. Затем, по методу наименьших квадратов вычисляется ошибка относительно истинной глубины.

Рис. 16 : Обработка исходного изображения (слева) моделями MiDaS и ZoeDepth: разница между относительной картой

Измерения МНК будут проводиться на изображениях открытых (рис. 17) и закрытых (рис. 19) пространств. Изображения с истинной глубиной были предоставлены наборами данных Virttual KITTI 2 [20] (синтетический набор данных, демонстрирующих открытые пространства) и Hypersim [22] (набор данных, состоящий из изображений и видео 3D-сцен, созданных с помощью движка Unreal Engine, включающий в себя 100 различных сцен, таких как интерьеры, городские улицы и природные ландшафты - для теста будет исопльзоваться закрытые помещения).

По результатам сравнения (см. рис. 18), видно, что точность оценки глубины ZoeDepth отличается от точности оценки MiDaS в пределах погрешности - 86.4% у ZoeDM12-NK против 87.0% у DPT-Large относительно набора данных Virtual KITTI v2. Однако в сравнении с данными из Hypersim DPT-Large показывает себя значительно хуже - 75.2% точности, в то время как ZoeDepth определяет глубину с точностью до 99.2%. Это говорит о том, что ZoeD-M12-NK лучше сохраняет свои характеристики между разными тестовыми наборами данных, что и заявляли авторы ZoeDepth в своей работе.

Таким образом, жертвуя незначительным процентом точности оценки, ZoeDepth предлагает модель, которая универсальна относительно глубинных характеристик изображения, и при этом вычисляет метрическую (а не относительную) карту глубины. Эта особенность позволяет использовать модель для широкого круга задач, связанных с измерением глубины, таких как распознавание объектов на изображениях, разработка алгоритмов автономного вождения и другие приложения, где требуется точное знание расстояния до объектов.

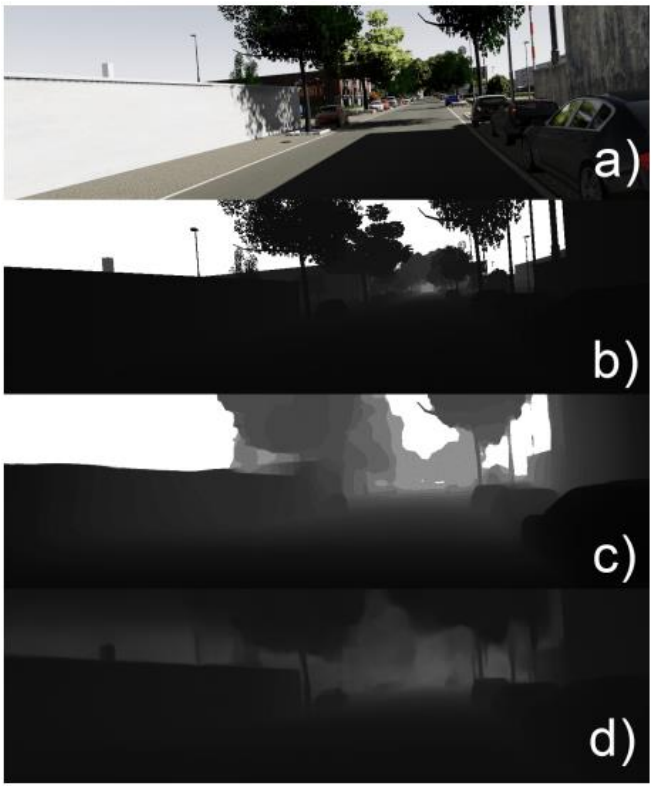


Рис. 17 Обработка исходного изображения на открытой местности (a) с помощью MiDaS (c - результат в негативе) и ZoeDepth (d). Приведена истинная глубина (b) для сравнения.

	Virtual KITTI v2		Hypersim	
	δ	MSE	δ	MSE
DPT-Large	64.01%	**0.1295**	50.18%	0.2482
ZoeD-M12-NK	63.13%	0.1359	91.12%	**0.0079**

Рис. 18 Оценка эффективности оценки глубины моделей DPT-Large под эгидой MiDaS и ZoeD-M12-NK, одной из моделей ZoeDepth. Точность - δ (%), и МНК - лучший результат выделен жирным.

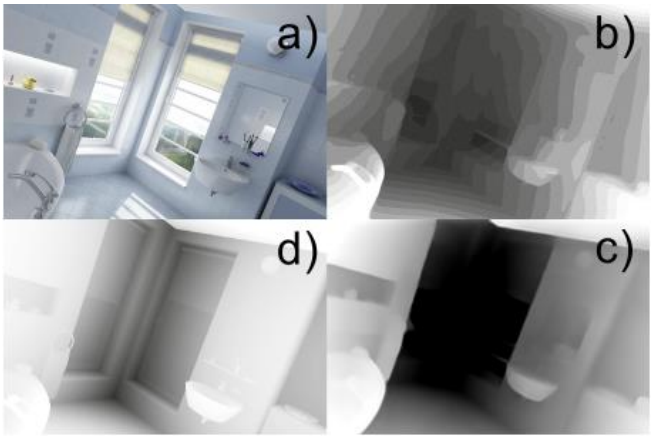


Рис. 19 бработка исходного изображения в закрытом по мещении (a) с помощью MiDaS (b - результат в негативе) и ZoeDepth (c). Приведена истинная глубина (d) для сравнения.

V. Заключение

В рамках данной статьи были описаны основные принципы работы моделей MiDaS и ZoeDepth, их архитектура, использованные наборы данных и функции потерь. По результатам сравнения можно сделать вывод, что современным моделям приходится бороться со множеством проблем - в т.ч. ограничениями на специфику обучающих наборов, а также компромиссами, связанными с попытками их обойти. Современные методы оценки глубины далеки от идеала, и всё ещё не способны предоставлять действительно качественные и метрически корректные результаты.

В настоящее время перспективным вопросом является решение проблемы универсальности модели, а также построение наиболее эффективных обучающих наборов данных. При разработке MiDaS был предложен набор данных, полученный на основе стереоскопических фильмов.

Качество такого набора данных остаётся под вопросом, поскольку проверочная глубина изображения вычислена искусственно (с помощью карт смещений), а не измерена традиционными способами.

С развитием в сфере трёхмерных технологий, наиболее выгодным стоит рассматривать формирование наборов данных на основе трёхмерных CGI-сцен. Возможность получать искусственные фотореалистичные изображения с минимальными затратами [23] может открыть возможность для создания искусственного обучающего набора данных, не отличимого от набора реальных фотографий, с идеальной истинной глубиной изображения.

Рассматриваемая перспектива лишь одна из многих, что предвещает множество открытий в сфере оценки глубины по монопоследовательности, и не только.

Список литературы

[1] K.V. Anokhin и др. “AI for Science and Science for AI”. В: Voprosy Filosofii (2022), с. 93—105.

[2] G. Brassard, P. F. Hoyer и A. Tapp. “Quantum algorithm for the collision problem”. В: Springer eBooks (2016), с. 891—921. DOI: https : / / doi . org / 10 . 1007 / 978-1-4939-2864-4_304.

[3] G. Brassard, P. F. Hoyer и A. Tapp. “Quantum algorithm for the collision problem”. В: Springer eBooks (2016), с. 891—921. DOI: https : / / doi . org / 10 . 1007 / 978-1-4939-2864-4_304.

[4] K. Xian и др. ““Monocular relative depth perception with web stereo data supervision”. В: CVPR (2018).

[5] Zhengqi Li и Noah Snavely. “Megadepth: Learning singleview depth prediction from internet photos”. В: CVPR (2018).

[6] C. Wang и др. “Web stereo video supervision for depth prediction from dynamic scenes”. В: 3DV (2019).

[7] Y. Kim и др. “Deep monocular depth estimation via integration of global and local predictions”. В: IEEE Transactionson Image Processing 27.8 (2018).

[8] Z. Li и N. Snavely. “MegaDepth: Learning single-view depth prediction from Internet photos”. В:CVPR (2018).

[9] N. Silberman и др. “Indoor segmentation and support inference from RGBD images”. В: ECCV (2012).

[10] Weihao Yuan и др. “New crfs: Neural window fullyconnected crfs for monocular depth estimation”. В: Springer eBooks (2022).

[11] Moritz Menze и Andreas Geiger. “Object scene flow for autonomous vehicles”. В: CVPR (2015).

[12] Rene Ranftl, Alexey Bochkovskiy и Vladlen Koltun. “Vision transformers for dense prediction”. В: IEEE/CVF (2021), с. 12179—12188.

[13] Hangbo Bao, Li Dong и Furu Wei. “Beit: BERT pretraining of image transformers”. В: CoRR (2021).

[14] Shariq Farooq Bhat, Ibraheem Alhashim и Peter Wonka. “Localbins: Improving depth estimation by learning local distributions”. В: Springer (2022), с. 480—496.

[15] PyTorch 2.1 documentation. Softmax. URL: https:// pytorch . org / docs / stable / generated / torch . nn.Softmax.html.

[16] W. Chen и др. “ingle-image depth perception in the wild”. В: NIPS (2016).

[17] T. Sch¨ops и др. “A multi-view stereo benchmark with high-resolution images and multi-camera videos”. В: CVPR (2017).

[18] D. J. Butler и др. “A naturalistic open source movie for optical flow evaluation”. В: ECCV (2012).

[19] J. Sturm и др. “A benchmark for the evaluation of RGBD SLAM systems”. В: IROS (2012).

[20] Yohann Cabon, Naila Murray и Martin Humenberger. “Virtual kitti 2”. В: (2020).

[21] Youngjung Kim и др. “Deep monocular depth estimation via integration of global and local predictions”. В: IEEE transactions on Image Processing 27.8 (2018), с. 4131—4144.

[22] Apple. Hypersim: A Photorealistic Synthetic Dataset for Holistic Indoor Scene Understanding. URL: https : //github.com/apple/ml-hypersim.

[23] State of Unreal GDC 2023. Unreal Engine 5.2 Tech Demo Full Presentation. URL: https://youtu.be/ Dj60HHy-Kqk.

Непрерывное распознавание языка жестов

К. А. Вершинин
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
evenmares@gmail.com

К. В. Башурина
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
bashksusha@mail.ru

***Аннотация*— Непрерывное распознавание языка жестов является актуальной областью исследований, нацеленной на разработку систем, способных интерпретировать жесты и переводить их в понятный формат для взаимодействия между человеком и компьютерной технологией. Данная статья описывает два метода компьютерного зрения, направленных на распознавание и классификацию жестовых движений из видео. Рассматриваемые нейронные сети Corrnet и Twostream-SLR имеют разные реализации архитектур, которые позволяют обрабатывать последовательности жестов для точного и эффективного их распознавания. Corrnet ориентирована на использование корреляций между парами кадров для выявления шаблонов, в то время как Twostream-SLR представляет собой сеть с двумя потоками, интегрирующими оптический поток и пространственные признаки для более полного анализа видео.**

Ключевые слова — компьютерное зрение, непрерывное распознавание жестов, распознавание жестов в реальном времени, классификация жестовых движений, corrnet, twostream-SLR.

I. Введение

Непрерывное распознавание языка жестов — это процесс анализа и интерпретации жестовых движений, используемых для коммуникации и передачи информации, применяемых в виде языка жестов (Sign Language Recognition – SLR) или других систем жестовой коммуникации [1, 2].

Основная суть этого процесса заключается в создании систем, способных в реальном времени распознавать и понимать жесты, выполняемые человеком, и преобразовывать их в соответствующий текст, речь или другой формат, понятный компьютерам или другим устройствам [3].

Процесс непрерывного распознавания языка жестов включает в себя использование различных методов компьютерного зрения, обработки сигналов и машинного обучения для анализа жестовых последовательностей, захваченных с помощью видеокамер или других сенсоров. Эти методы позволяют системе распознавать и классифицировать различные жесты, учитывая их форму, движение, положение в пространстве и контекст их использования. Целью непрерывного распознавания языка жестов является создание удобных и эффективных средств взаимодействия между людьми, использующими жесты для коммуникации, и технологией. Это может помочь людям с нарушениями слуха или речи в общении с другими людьми или управлении компьютерами и устройствами. Также это имеет потенциал для применения в различных областях, таких как образование, виртуальная и дополненная реальность, медицинские технологии, а также в других сферах, где жестовая коммуникация может быть эффективным средством взаимодействия [4,5].

Непрерывное распознавание языка жестов также стремится к созданию систем, способных обрабатывать непрерывные потоки жестов, обеспечивая более естественное и мгновенное взаимодействие между человеком и технологией. Это означает способность системы распознавать и интерпретировать жесты в реальном времени без значительной задержки [6].

Суть непрерывного распознавания жестов заключается в обработке динамической и вариативной природы жестовых движений. Она включает в себя учет не только формы и конкретного жеста, но и его контекста, такого как мимика лица, поза тела и другие невербальные сигналы, которые могут дополнять и уточнять значение жеста [7].

Для реализации рассматриваемых сетей исследователи и инженеры используют разнообразные технологии, включая компьютерное зрение, машинное обучение, глубокие нейронные сети, а также методы обработки сигналов. Эти методы позволяют системам "учиться" распознавать и адаптироваться к различным жестам и их вариациям. Инновации в этой области направлены на создание более точных и быстрых систем, способных распознавать и интерпретировать жестовые коммуникации с высокой точностью и надежностью. Успешное развитие и применение таких систем может значительно улучшить доступ к коммуникации и технологиям для людей с различными потребностями и ограничениями, а также дать новые возможности в области взаимодействия человека с компьютерными системами.

Данная технология используется в работе виртуальных ассистентов, для сурдоперевода, в приложениях с дополненной реальностью и развлекательных сервисах.

II. Наборы данных

Для обучения и тестирования работы сетей использовались два набора открытых данных. Оба набора данных представляют из себя видеоматериалы с разнообразными жестовыми последовательностями, собранными при одинаковых условиях и с удаленным фоном. Для работы данных сетей важны не только жесты рук, но и мимика лица, позы тела и движения головы.

A. RWTH-PHOENIX-Weather 2014

В течение трех лет (2009–2011) были записаны ежедневные выпуски новостей и прогноз погоды немецкого общественного телеканала PHOENIX с интерпретацией речи ведущих в жестовую запись [8]. В настоящее время только прогнозы погоды из подмножества 386 выпусков были расшифрованы и сохранены в формате аннотации. Расшифровки были выполнены глухими и слабослышащими носителями немецкого языка жестов. Кроме того, устный прогноз погоды на немецком языке был расшифрован полуавтоматическим способом с использованием системы распознавания речи RASR. Пример оригинального видео вместе с сурдопереводчиком представлено на рисунке 1.

Рисунок 1 – пример исходного видео вместе с переводчиком на жестовый из набора данных RWTH-PHOENIX-Weather 2014

Жестовый перевод записывался стационарной цветной камерой, установленной перед сурдопереводчиками. Переводчики носят темную одежду на фоне искусственного серого фона с цветовым переходом. Все записанные 40 000 видео воспроизводятся со скоростью 25 кадров в секунду, а размер кадров составляет 210 на 260 пикселей. В каждом кадре отображается только окно переводчика.

B. RWTH-PHOENIX-Weather 2014-T

Рассматриваемый второй набор данных [9] можно рассматривать как более расширенный и качественнее проработанный датасет RWTH-PHOENIX-Weather 2014. Ежедневные выпуски новостей и прогнозов погоды были также получены с немецкой общественной телестанции PHOENIX с сурдопереводом. Различие заключается в том, для рассматриваемого датасета использовали автоматическое распознавание речи с ручной очисткой для расшифровки оригинальной немецкой речи. Таким образом, этот корпус позволяет обучать сквозные системы сурдоперевода с видеовхода на языке жестов на разговорный язык. Данный датасет имеет меньшее количество видео, около 20 000.

III. Нейросетевые архитектуры

A. CorrNet

1. Формулировка задачи и архитектура сети

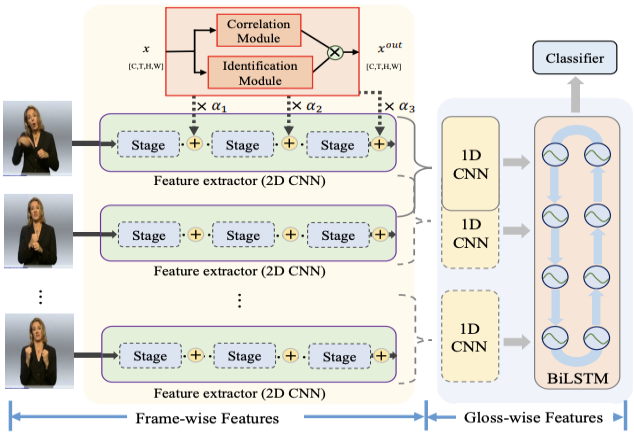


Рисунок 2 – архитектура сети CorrNet

Как показано на рисунке 2, основа модели состоит из экстрактора признаков (2D сверточные сети), 1D сверточные сети, двойной LSTM и классификатора (полносвязный слой) для предсказания. Дано видео с контентом на языке жестов с T входными кадрами $x = \{x_t\}_{t=1}^{T} \in \mathcal{R}^{T\times 3\times H_0\times W_0}$ модель CorrNet направляет на перевод входного видео в серию глоссов $y = \{y_i\}_{i=1}^{N} \in \mathcal{R}^{T\times d}$ для выражения предложения, причем N обозначает длину последовательности. В частности, экстрактор признаков сначала преобразует входные кадры в признаки по кадрам $v = \{v_t\}_{t=1}^{T} \in \mathcal{R}^{T\times d}$. Затем одномерная сверточная сеть и билинейная LSTM выполняют краткосрочное и долгосрочное временное моделирование на основе этих извлеченных визуальных представлений, соответственно. В результате классификатор использует широко распространенную потерю CTC ошибку, чтобы прогнозировать вероятность появления целевой последовательности глосса $p(y|x)$ [10].

CorrNet модель обрабатывает входные кадры независимо друг от друга, не учитывая взаимодействие между последовательностями кадров. Данная сеть представляет модуль корреляции и модуль идентификации для определения траекторий движения тела в соседних кадрах. На рисунке 2 показан пример общего экстрактора признаков, состоящего из нескольких этапов. Предлагаемые два модуля располагаются после каждого этапа, их выводы перемножаются по элементам и добавляются к исходным признакам с помощью обучающего коэффициента α. α контролирует вклады предложенных модулей и инициализируется нулем, чтобы вся модель сохраняла свое первоначальное поведение. Модуль корреляции вычисляет корреляционные карты между последовательностями кадров, чтобы зафиксировать траектории всех пространственных пятен. Модуль идентификации динамически находит и подчеркивает траектории тела, встроенные в эти корреляционные карты. Выходы модулей корреляции и идентификации перемножаются для усиления межкадровых корреляций.

2. Корреляционный модуль

Язык жестов в основном передается с помощью ручных компонентов (жесты рук, ладони) и неручных компонентов (мимика лица, движения головы и позы тела). Однако эти информативные части тела, то есть руки или лицо, смещены в соседних кадрах. Предлагается вычислять корреляционные карты между соседними кадрами для определения траекторий движения тела.

Каждый кадр должен быть представлен как 3D тензор вида $C \times H \times W$, где C – количество каналов, а $H \times W$ обозначает пространственный размер. Получив участок признаков $p_t(i,j)$ на текущем кадре x_t, рассчитывается сродство между патчем $p_{t+1}(i',j')$ в соседнем кадре x_{t+1}, где (i, j) это пространственное расположение патча. Чтобы ограничить вычисления, можно уменьшить размер характерного патча до

минимума, то есть до пикселя. Сродство между $p(i,j)$ и $p_{t+1}(i',j')$ вычисляется векторным умножением как

$$A(i,j,i',j') = \frac{1}{C}\sum_{c=1}^{C}(p_t^c(i,j) \cdot p_{t+1}^c(i',j'))$$

Для пространственного местоположения (i,j) в x_t, (i,j) часто ограничивается в окрестности $K \times K$ в x_{t+1}, чтобы устранить пространственное несоответствие. Архитектура корреляционного оператора представлена на рисунке 3. Таким образом, для всех пикселей в x_t, корреляционные карты представляют собой тензор размера $H \times W \times K \times K$ может быть задано как меньшее значения для сохранения семантической согласованности, или большее значение, чтобы обратить внимание на удаленные информативные области.

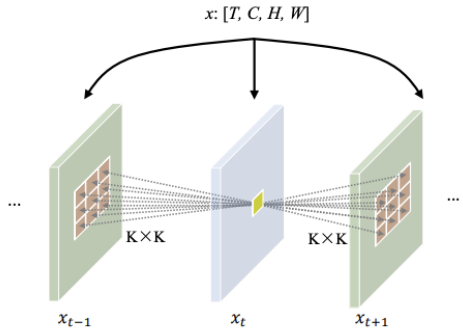


Рисунок 3 – иллюстрация корреляционного оператора

Учитывая карту корреляции между пикселем и его соседями в соседнем кадре x_{t+1}, мы ограничиваем его в диапазоне (0, 1), чтобы измерить их семантическое сходство путем передачи $A(i,j,i',j')$ через сигмоидную функцию. Далее вычитается 0.5 из результатов, чтобы подчеркнуть информативные области с положительными значениями и подавляем избыточные области с отрицательными значениями.

После определения траекторий между соседними кадрами включаются локальные временные перемещения в текущий кадр x_t. В частности, для пикселя в x_t, его траектории агрегируются из его $K \times K$ соседей в соседних кадрах x_{t+1}, умножая их характеристики на соответствующее сродство как

$$T(i,j) = \sum_{i',j'} A'(i,j,i',j') * x_{t+1}(i',j')$$

В этом смысле каждый пиксель может знать о своих траекториях в последовательных кадрах. Далее агрегируются двунаправленные траектории из x_{t-1} и x_{t+1}, и добавляются обучаемый коэффициент β для измерения важности двунаправленных траекторий.

3. Идентификационный модуль

Модуль корреляции вычисляет карты корреляции между каждым пикселем и его соседями $K \times K$ в соседних кадрах x_{t-1} и x_{t+1}. Однако, поскольку не все области являются критическими для выражения признака, в текущем кадре x_t следует выделять только информативные области, несущие траектории движения тела. Траектории фона и шума должны быть подавлены. Для этого используется модуль идентификации для динамического выделения этих информативных пространственных областей. В частности, поскольку информативные области, такие как рука и лицо, смещены в соседних кадрах, модуль идентификации использует тесно коррелирующие локальные пространственно-временные особенности чтобы решить проблему несоответствия и найти информативные области.

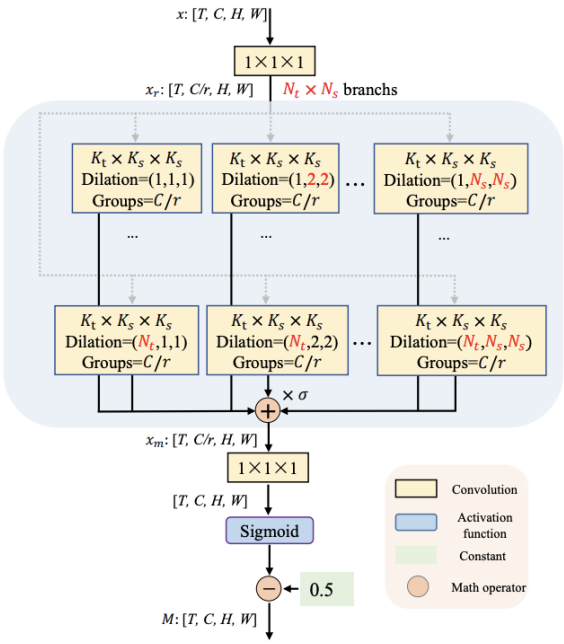


Рисунок 4 – иллюстрация идентификационного модуля

Как показано на рисунке 4, модуль идентификации сначала проецирует признаки $x \in \mathcal{R}^{T\times C\times H\times W}$ в $x_r \in \mathcal{R}^{T\times C/r\times H\times W}$ со сверткой $1 \times 1 \times 1$ для уменьшения объема вычислений, на коэффициент уменьшения каналов r, который по умолчанию равен 16.

Поскольку информативные области, например руки и лицо, не точно выровнены в соседних кадрах, необходимо рассматривать большую пространственно-временную окрестность для идентификации этих особенностей. Вместо прямого использования большого трехмерного пространственно-временного ядра, мы представляем многомасштабную парадигму разложив его на параллельные ветви с прогрессивной скоростью расширения, чтобы уменьшить количество необходимых вычислений и увеличить емкость модели.

В частности, как показано на рисунке 4, при одинаково малом базовом ядре свертки $K_t \times K_s \times K_s$, мы используем несколько сверток с увеличением их скорости по пространственному и временному измерениям одновременно. Пространственное и временное измерения находятся в пределах (1, N_s) и (1, N_t), соответственно, в результате чего получается $N_s \times N_t$

ветвей. Для каждой ветки используются групповые свертки для уменьшения параметров и вычислений. Признаки из разных ветвей перемножаются с обучаемыми коэффициентами $\{\sigma_1, \ldots, \sigma_{Ns\times Nt}\}$, чтобы для контроля их важности, а затем добавляются для смешивания информации из ветвей различных пространственно-временных рецептивных полей.

B. *TwoStream Network*

1. Формулировка задачи и архитектура сети

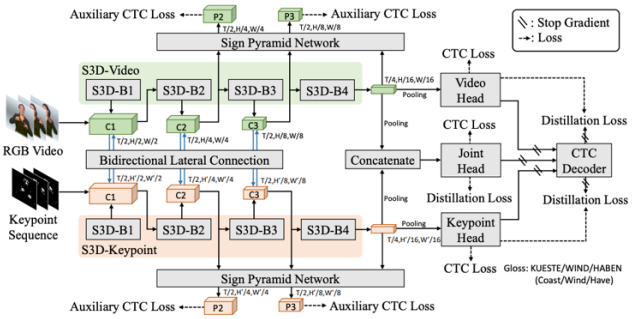


Рисунок 5 – архитектура сети TwoStream-SLR

На рисунке 5 показан вид сети TwoStream SLR, которая состоит из пяти частей: видеокодера, кодер последовательности ключевых точек, совместная шапка, двунаправленного связующего модуля, двух пирамидальных сетей, для моделирования RGB-видео и последовательностей ключевых точек [11].

2. Видео энкодер

В качестве видеоэнкодера используется легковесная головная сеть S3D. Только первые четыре блока S3D, поскольку целью является извлечение плотных представлений во временном измерении. Подается каждое видео размера $T \times H \times W$ в энкодер для извлечения его характеристик. В данном случае T обозначает канал, H и W – высоту и ширину видео. По умолчанию, H и W равны по 224. Выходной признак последнего блока S3D пространственно объединяется в размер $T/4 \times 832$ перед подачей в головную сеть. Целью головной сети является дальнейший захват временного контекста. Она состоит из временного линейного слоя, слоя пакетной нормализации, слоя ReLU и темпорального сверточного блока, который содержит два сверточных слоя ядром размером 3, шагом 1, линейного трансляционного слоя и слоя ReLU. Выходной признак, названный представлением глосса, имеет размер $T/4 \times 512$. Затем для выделения глосса применяется линейный классификатор и функция SoftMax для извлечения вероятностей глосса на уровне кадра. Наконец, используется связанный темпоральный классификационную ошибку для оптимизации видеоэнкодера.

3. Энкодер ключевых точек

Для моделирования последовательностей ключевых точек используется предобученная сеть HRNet на датасете COCO-WholeBody, используемая для генерации 42 ключевых точек руки, 68 ключевых точек лица, охватывающих рот, глаза и контур лица, а также 11 ключевых точек верхней части тела, охватывающих плечи, локти и запястья на каждом кадре. Эмпирическим путем обнаружено, что использование только подмножества из 26 ключевых точек лица (10 ключевых точек для рта и 16 для других частей) дает хорошие результаты, экономя при этом вычислительные ресурсы. Всего используется 79 ключевых точек. В данной сети используются тепловые карты для представления ключевых точек.

4. Двунаправленный связующий модуль

Чтобы объединить информацию двух потоков, мы предлагаем латеральную связь, которая изучается в распознаваний действий и обнаружении объектов. Латеральное соединение реализуется как операция поэлементного сложения двух карт признаков с одинаковым разрешением. Применяются латеральные связи к признакам (C_1, C_2, C_3), сгенерированными первыми тремя блоками (B_1, B_2, B_3) двух backbone S3D. Поскольку пространственные разрешения промежуточных признаков, извлеченных из двух потоков различны, используется свертка с пространственной разверткой и транспонированная свертка для выравнивания их пространственных разрешений.

5. Совместная головная сеть и ансамбль

Видеоэнкодер и энкодер ключевых точек имеют свои собственные головные сети. Чтобы полностью раскрыть потенциал архитектуры с двумя энкодерами, представляется дополнительная объединенная головная сеть, которая принимает на вход конкатенацию выходов двух сетей S3D в качестве входных данных. Архитектура данной сети такая же, как у головной сети видеоэнкодера и энкодера ключевых точек. Совместная головная сеть также контролируется CTC-ошибкой. Далее идет усреднение по кадрам вероятности глосса, предсказанные видеоэнкодером, энкодером головных точек и совместной сетью, и подается на декодер CTC для генерации последовательности глосса. Эта стратегия позднего ансамбля объединяет результаты нескольких потоков и улучшает результаты по сравнению с предсказаниями в одном потоке.

6. Символьная пирамидальная сеть

Чтобы лучше улавливать глоссы разных временных интервалов и эффективно контролировать неглубокие слои для обучения значимым представлениям, строится сеть пирамидальных знаков (SPN) со вспомогательным контролем для двойного визуального энкодера. Архитектура данной сети показана на рисунке 6.

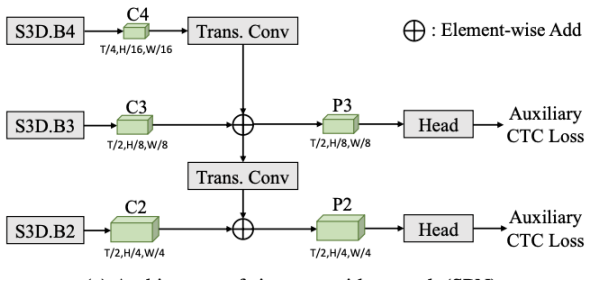


Рисунок 6 – архитектура символьной пирамидальной сети

В частности, обозначим выходы трех последних блоков опорной сети S3D как C_2, C_3 и C_4 соответственно. Используется операция поэлементного сложения для объединения признаков, извлеченных из соседних блоков S3D, и слитые признаки обозначаются как P_2 и P_3. Используется транспонированная свертка, чтобы согласовать временные и пространственные размеры двух карт признаков перед поэлементным сложением, затем две отдельные головные сети с той же архитектурой, что и в двойном энкодере, применяются к P2 и P3 для извлечения вероятностей глосса на уровне кадра. Аналогично, для обеспечения вспомогательного контроля используется CTC-лосс. Без потери общности используются две независимых символьные пирамидальные сети для видео и потока ключевых точек.

7. Самодисцилляция на уровне кадров

Существующие наборы данных предоставляют аннотации глоссов только на уровне предположений, где временные границы глоссов не обозначены. Таким образом, CTC-лосс широко используется для такого слабого контроля. Однако после хорошей оптимизации визуальный энкодер способен генерировать вероятности глосса по кадрам, из которых можно оценить приблизительную временную границу глоссов. Такие результаты позволяют использовать предсказанные вероятности глоссов по кадрам в качестве псевдоцелей для того, чтобы обеспечить дополнительный тонкий контроль в дополнение к грубому CTC. В соответствии с двухпотоковой архитектурой, используются усредненные вероятности глосса из трех головных сетей в качестве псевдоцелей для управления обучением каждого потока. Формально, происходит минимизация расхождение между псевдоцелями и предсказаниями трех головных сетей [12].

8. Функция ошибок

Общие потери TwoStream-SLR состоят из трех частей: 1) CTC-лосс на выходах видеоэнкодера ($\mathcal{L}_{CTC}^{V}$), энкодера ключевых точек ($\mathcal{L}_{CTC}^{K}$) и совместной головной сети ($\mathcal{L}_{ACTC}^{V}$); 2) Вспомогательные CTC-лоссы ($\mathcal{L}_{ACTC}^{V}$) и ($\mathcal{L}_{ACTC}^{K}$), применяемые на выходах двух знаковых пирамидальных сетей; 3) Потери при дистилляции ($\mathcal{L}_{dist}$). Тем самым формируются потери при распознавании следующим образом:

$$\mathcal{L}_{SLR} = \mathcal{L}_{CTC}^{V} + \mathcal{L}_{CTC}^{K} + \mathcal{L}_{CTC}^{J} + \lambda_V \mathcal{L}_{ACTC}^{V} + \lambda_K \mathcal{L}_{ACTC}^{K} + \mathcal{L}_{Dist}$$

Где λ_V и λ_K обозначают веса потерь вспомогательного CTC-лосса видео и лосса ключевых точек потока. После обучения сеть TwoStream-SLR способна предсказывать последовательность глосса путем усреднения предсказаний от трех головных сетей.

IV. Сравнение

Рассмотрим численные результаты сравнения сетей, представленные в таблицах 1 и 2.

В качестве метрики сравнения используется коэффициент ошибок слов (WER), который определяется как минимальное суммирование операций подстановки, вставки и удаления для преобразования предсказанное предложения в эталонное. То есть, чем ниже WER, тем выше качество.

$$WER = \frac{S + D + I}{T},$$

Где S – количество замененных слов, D – количество удаленных слов, I – количество дополнительных слов, а T – количество истинных слов в эталонной фразе.

Таблица 1. Результаты работы сетей на наборе данных Phoenix-2014

	WER (%)
CorrNet	18,8
TwoStream-SLR	18,4

Таблица 2. Результаты работы сетей на наборе данных Phoenix-2014-T

	WER (%)
CorrNet	18,9
TwoStream-SLR	17,7

По результатам видно, что у модели TwoStream-SLR значения WER-метрики меньше, соответственно, работает качественнее.

V. Заключение

Непрерывное распознавание языка жестов представляет собой важную область развития технологий, ориентированную на создание систем, способных улучшить коммуникацию между людьми, использующими язык жестов, и технологическими устройствами. Эта технология открывает перед нами возможности для создания более эффективных средств взаимодействия и управления, помогая людям с нарушениями слуха или речи, а также находя применение в различных областях, таких как образование, медицина и развлечения.

Используя компьютерное зрение, методы машинного обучения и обработки сигналов, системы непрерывного распознавания языка жестов стремятся к созданию более точных, быстрых и адаптивных моделей. Их цель – предоставить более естественные и мгновенные способы взаимодействия человека с технологией. Развитие таких систем имеет потенциал улучшить доступность коммуникации и технологии для всех, а также создать новые

платформы и взаимодействия человека с компьютерными системами.

Однако, несмотря на достигнутые успехи, вопросы точности, скорости обработки и адаптации к разнообразию жестов остаются вызовом для исследователей и инженеров. Дальнейшие исследования и инновации в области непрерывного распознавания языка жестов не только помогут улучшить технологию распознавания, но и увеличат ее доступность и широкий спектр применения, обогащая жизнь многих людей по всему миру.

Литература

[1] Iliukhin, S. *et al.* (2019) ‘A method for spatially weighted image brightness normalization for face verification’, *Eleventh International Conference on Machine Vision (ICMV 2018)* [Preprint]. doi:10.1117/12.2522922.

[2] Pazychev, D.B. and Sadekov, R.N. (2020) ‘Simulation of INS errors of various accuracy classes’, *2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins43215.2020.9133869.

[3] L. Hu, L. Gao, Z. Liu, W. Feng, Continuous Sign Language Recognition with Correlation Network, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2023.

[4] Bikmaev, R.R. *et al.* (2019) ‘Improving the accuracy of supporting mobile objects with the use of the algorithm of complex processing of signals with a monocular camera and Lidar’, *2019 26th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)* [Preprint]. doi:10.23919/icins.2019.8769360.

[5] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N.Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in anAutonomous Tram to Predict the Behaviour of Vehicles in the RoadScene," 2022 29th Saint Petersburg International Conference onIntegrated Navigation Systems (ICINS), Saint Petersburg, RussianFederation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[6] Savelyev, B & Solodov, S & Tropin, D. (2021). Formalizing andsecuring relationships on multi-task metric learning for IoT-basedsmart cities. Journal of Physics: Conference Series. 2094. 032062.10.1088/1742-6596/2094/3/032062.

[7] Y. Chen, F. Wei, X. Sun, Z. Wu, S. Lin, A Simple Multi-Modality Transfer Learning Baseline for Sign Language Translation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2022: pp. 5120–5130.

[8] O. Koller, J. Forster, and H. Ney. Continuous sign language recognition: Towards large vocabulary statistical recognition systems handling multiple signers. Computer Vision and Image Understanding, volume 141, pages 108-125, December 2015.

[9] Necati Cihan Camgöz, Simon Hadfield, Oscar Koller, Hermann Ney, Richard Bowden, Neural Sign Language Translation, IEEE Conf. on Computer Vision and Pattern Recognition, Salt Lake City, UT, 2018.

[10] https://github.com/hulianyuyy/CorrNet

[11] https://github.com/FangyunWei/SLRT/blob/main/TwoStreamNetwork/docs/TwoStream-SLR.md

[12] Y. Chen, R. Zuo, F. Wei, Y. Wu, S. Liu, B. Mak, Two-Stream Network for Sign Language Recognition and Translation, NeurIPS. (2022).

Исследование возможности классификации картин при помощи компьютерного зрения

Я. О. Кудинов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2100108@edu.misis.ru

***Аннотация*— в настоящее время широко распространяются нейронные сети, в том числе для генерации изображений. В искусстве и в картинах в частности довольно тяжело разбираться на достаточном уровне чтобы распознать подделку. Целью данного исследования является проверка возможности сделать классификатор картин с помощью нейронной сети. Картины имеют множество параметров, по которым их можно разделять на группы, в данной работе будет рассмотрена классификация по авторам, однако жанры также имеют высокую актуальность. Потенциально нейронные сети смогут указывать на поддельные работы а также показывать информацию об интересующих человека объектах. В работе рассматривается сверточная нейронная сеть из библиотеки Keras, данные взяты из kaggle, куда их отфильтровали из wikiarts. Для оценки точности после обучения используются перемешанные картины из обучающего и валидационных наборов в силу ограниченного количества произведений у каждого автора.**

Ключевые слова — компьютерное зрение, глубокое обучение, картины, классификация картин, распознавание картин

I. Введение

В наше время искусство является предметом интереса все большего количества людей, однако не многие являются достаточно квалифицированными для точного определения какой из авторов представлен перед ними, не говоря уже о ситуации когда нужно определить подделку. Кроме того у многих есть потребность избегания лишних человеческих контактов, что затрудняет поиск информации по заинтересовавшим человека объектам искусства. Для решения подобных проблем хорошо подходят нейронные сети которые могут определять малейшие детали в картинах, а также быстро выдавать большой объем информации[1,2].

Основной сложностью работы при создании классификатора картин является возможность присутствия помех на практике так как при съемке картины в помещении или на улице, свет может усложнить определение какого либо стиля или автора.

Распознавание и идентификация картин поможет избежать подделок и получить справочную информацию по объекту. Для классификации картины по автору определяется стилистика характерная для данного конкретного художника, в которую входит цветовая гамма, закономерности в геометрических формах. Если рассматривать классификацию по стилям то тогда используется характерное определение для заключения ряда закономерностей.

В задачах подобного типа хорошо проявили себя методы глубокого обучения, как в сфере классификации так и имитации различных стилей[3].Эти методы далеко не ограничены этими задачами и используются во многих областях таких как например – шифрование изображений[4], медицина [5], автопилоты[6,7,8] и ряд других сфер[9,10,11].В силу высокой производительности генеративных нейронных сетей, существует огромное количество моделей для создания картин, а для того чтобы генерировать картины в определенном стиле нужно сперва обучить эту модель данному стилю, соответственно в результате получается внутренний классификатор по стилям, но также зачастую и по художникам[12]. В силу простоты использования подобных моделей и их большого количества, большой интерес представляет процесс создания классификатора позволяющего разделять картины по определенному количеству параметров.

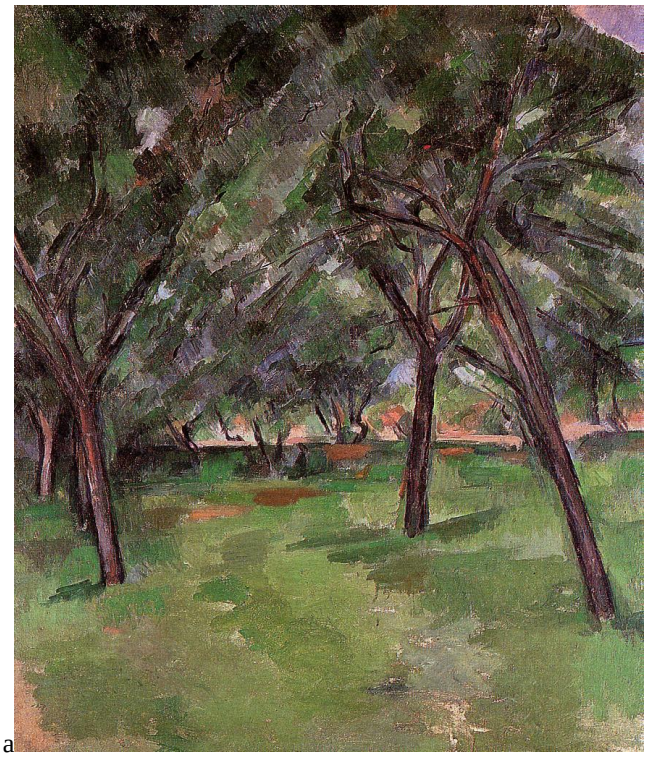
Подходы, основанные на обучении, особенно те, которые используют глубокое обучение, требуют больших объёмов аннотированных данных, в рамках искусства эта проблема является наиболее сложной, так как не у каждого автора есть достаточное количество работ для обучения нейронных сетей.Некоторые параметры, жанр например, использует работы большого количества авторов, что делает классификацию по авторам несколько сложнее.

II. Наборы данных

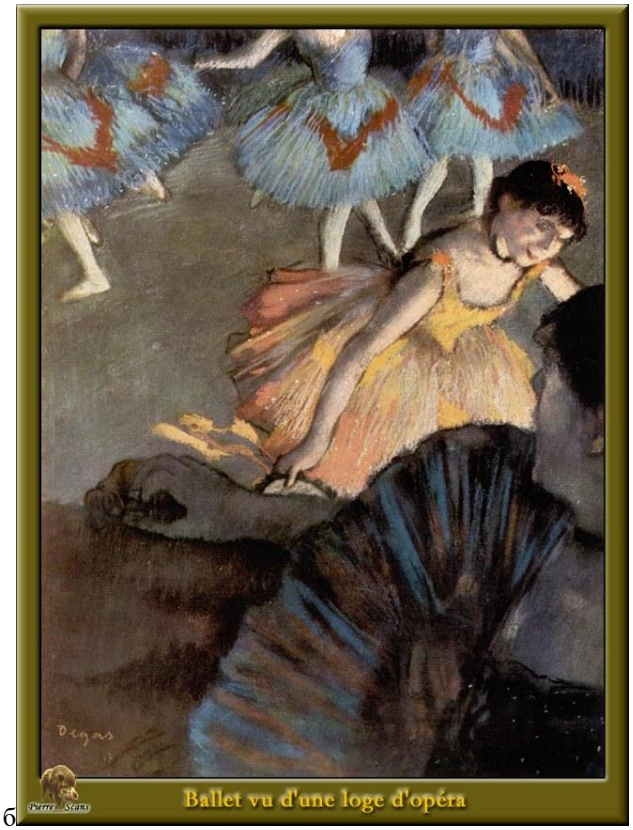
Для обучения и тестирования рассматриваемых в данной работе нейросетей использовался набор данных об импрессионистах художниках, состоящий из части набора данных wikiart, одного из двух крупнейших наборов данных по картинам.

A.*Kaggle*

Обширный набор данных о художниках импрессионистах был взят с сайта kaggle. Данный сборник содержит 2D изображения картин, разделенные на 10 классов: Camille Pissarro, Childe Hassam, Claude Monet, Edgar Degas, Henri Matisse, John Singer-Sargent, Paul Cézanne, Paul Gauguin ,Pierre-Auguste Renoir, Vincent van Gogh. Вообщей сложности набор содержит 5000 изображений по 500 на каждого художника. .



а

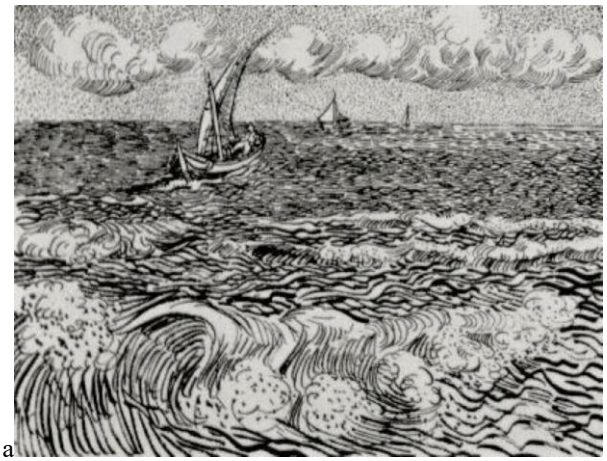


б

Рис.1 – примеры картин из набора данных, от двух художников

В. *Набор данных для проверки работоспособности*

Так как проверяется классификация по авторам а не стилям, для проверки использовались данные из обучающей выборки в перемешку с малым количеством работ отложенным для каждого автора, итоговый тестовый дата сет являет собой 90 картин каждого автора на которых модель не обучалась и 10 картин на которых происходило обучение.



а

б

Рис.2 – примеры картин из второго проверочного набора данных

III. НЕЙРОСЕТЕВАЯ АРХИТЕКТУРА

Сверточная нейронная сеть в Keras

Для решения данной задачи было решено использовать предварительно обученную модель поэтому в коде используется MobileNetV2 в качестве базовой модели, загружает предварительно обученные веса и добавляет сверху настраиваемое классификационное навершие. В целом архитектура MobileNetV2 содержит начальный уровень полной свертки с 32 фильтрами, за которым следуют 19 остаточных слоев узких мест, всего имея 53 слоя свертки[13]. Эта модель есть в библиотеке Keras и является сверточной нейронной сетью(CNN)[14]. Сверточные нейронные сети популярная архитектура глубокого обучения, Keras же предоставляет широкий спектр функциональный возможностей для создания и обучения сверточных нейронных сетей. MobileNetV2 это легковесная модель хорошо подходящая для использования на мобильных устройствах.

Далее представлена полная архитектура модели MobileNetV2:

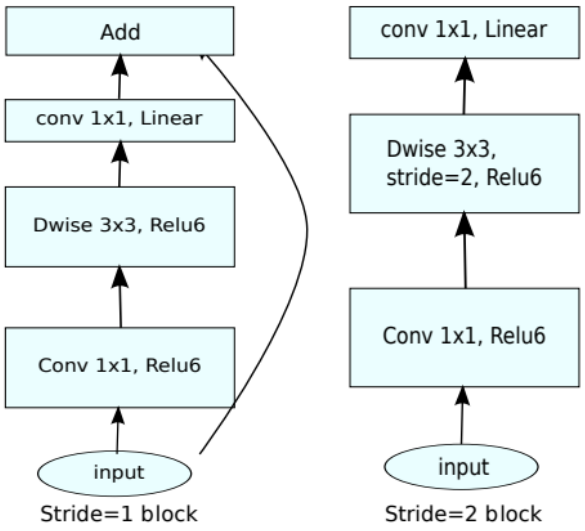


Рис.3 – архитектура сверточной мобильной сети MobilenetV2

Далее сперва стоит разобрать принцип работы сверточной нейронной сети, ведь MobilenetV2 является сверточной нейронной сетью, пусть и с рядом отличий. Начиная с первого слоя из библиотеки Keras.

Входной слой(Input Layer): входной слой определяет форму входных данных. Для изображений это будет (высота, ширина, каналы), где высота и ширина — это размеры входного изображения, а каналы представляют собой цветовые каналы (например, 3 для RGB). То Есть всего три параметра.

Сверточный слой(Convolutional Layer): сверточные слои отвечают за изучение пространственных иерархий объектов на основе входных данных. Параметры:

- filters: количество фильтров (ядер), которые будут применены к входным данным.
- kernel_size: Размер сверточных фильтров(размер ядра).
- strides: размер шага для операции свертки.
- activation: функция активации применяется после свертки.

Слой пулинга (например, MaxPooling2D): уменьшает пространственные размеры данных и помогает снизить вычислительные затраты и контролировать переобучение. MaxPooling применяется к каждому feature map независимо, чтобы уменьшить их пространственные размеры. Это помогает снизить вычислительную нагрузку и количество параметров в сети. MaxPooling делит входные данные на непересекающиеся прямоугольные области и для каждой области выводит максимальное значение. Обычно используемый размер пула равен (2, 2), что означает, что входные данные делятся на регионы 2x2, и сохраняется максимальное значение в каждом регионе. .Параметры: Pool_size: Размер окна пула.

Гладкий слой(Flatten Layer): Сглаживающий слой преобразует многомерный результат в одномерный вектор, подготавливая его для полносвязных слоев.Это преобразование необходимо для соединения сверточных слоев и слоев пула с последующими полносвязными (плотными) слоями. Он принимает входной тензор с несколькими измерениями (например, высота x ширина x каналы) и преобразует его в одномерный массив.

Плотный слой (Dense layer): плотные слои используются для изучения нелинейных комбинаций признаков и получения окончательных прогнозов. Параметры:

- units: количество нейронов в слое.
- activation: функция активации, примененная к слою.

Плотный слой (Dense layer): плотные слои используются для изучения нелинейных комбинаций признаков и получения окончательных прогнозов. Каждый нейрон в плотном слое связан с каждым нейроном в предыдущем слое, образуя полностью связную сеть. Выходные данные плотного слоя рассчитываются с использованием взвешенной суммы его входных данных, за которой следует функция активации.Выбор функции активации на плотном слое вносит в модель нелинейность, позволяя ей изучать сложные взаимосвязи. Общие функции активации включают «relu» (выпрямленная линейная единица), «sigmoid» и «softmax».

Выходной слой: выходной слой дает окончательные прогнозы. Параметры:

- unit: количество выходных нейронов (равно количеству классов в задачах классификации).
- activation: функция активации (например, «softmax» для многоклассовой классификации).

Таким образом, общую картину работы CNN можно описать следующим образом. Сверточные нейронные сети работают путем систематической обработки данных структурированной сетки, таких как изображения, через ряд специализированных слоев. Начальные сверточные слои используют обучаемые фильтры для извлечения локальных закономерностей и особенностей, фиксируя пространственные иерархии в данных. Функции активации, такие как выпрямленный линейный блок (ReLU), вводят нелинейность, позволяя сети изучать сложные сопоставления. Последующие слои объединения субдискретизируют пространственные измерения, сохраняя важную информацию и одновременно снижая вычислительную нагрузку. Слой Flatten преобразует многомерные выходные данные в одномерный вектор, облегчая подключение к полностью связанным (плотным) слоям. Эти плотные слои дополнительно изучают нелинейные комбинации функций, фиксируя глобальные закономерности и взаимосвязи. Последний выходной слой выдает прогнозы на основе изученных представлений с функциями активации, такими как softmax, для многоклассовой классификации. Обучение включает в себя корректировку параметров модели с помощью алгоритмов обратного распространения ошибки и оптимизации, чтобы минимизировать разницу между прогнозируемыми и фактическими результатами, руководствуясь функцией потерь. Методы регуляризации, такие как отсев и пакетная нормализация, используются для предотвращения переобучения. CNN часто используют трансферное обучение, используя предварительно обученные модели на больших наборах данных, чтобы извлечь выгоду из изученных функций.

MobileNetV2 представляет собой легкую архитектуру сверточной нейронной сети, оптимизированную для эффективности с точки зрения вычислительных затрат и

размера модели. Это достигается за счет использования нескольких ключевых принципов:

- Вместо традиционных сверток, которые работают со всем входным объемом, MobileNetV2 использует свертки, разделяемые по глубине. Это состоит из двух отдельных операций: глубинной свертки, за которой следует точечная свертка. Глубокая свертка выполняет упрощенную пространственную фильтрацию для каждого входного канала отдельно, снижая вычислительные затраты. Затем поточечная свертка объединяет отфильтрованные каналы в конечный результат. Такое разделение пространственной и канальной фильтрации значительно сокращает количество параметров и вычислений по сравнению со стандартными свертками. Инвертированные остатки с линейными узкими местами:
- MobileNetV2 представляет концепцию инвертированных остатков. В традиционных остаточных сетях (ResNets) короткое соединение обходит сверточные уровни. В MobileNetV2 короткое соединение применяется после облегченной свертки с разделением по глубине, образуя «инвертированный» остаточный блок.
- MobileNetV2 представляет гиперпараметр, называемый «множителем ширины» (альфа). Этот параметр управляет шириной сети путем масштабирования количества каналов на каждом уровне. Меньшая альфа уменьшает количество параметров и вычислений, делая сеть более компактной, но потенциально менее мощной.
- MobileNetV2 использует глобальную отделяемую по глубине свертку в конце сети. Эта операция фиксирует глобальные функции эффективным с точки зрения вычислений способом, позволяя сети иметь глобальное понимание входных данных.
- MobileNetV2 использует пропускаемые соединения, аналогично ResNets, чтобы облегчить градиентный поток во время обучения. Эти пропускаемые соединения применяются перед извилинами, разделяемыми по глубине, что облегчает процесс обучения[15].

IV. Проведение испытаний.

Перед тестированием работоспособности выбранной нейронной сети нужно ее обучить, для обучения нужно выбрать количество эпох. Эпоха – один из параметров непосредственно влияющих на процесс обучения и на итоговые результаты, она обозначает один проход через всю обучающую часть выбранного набора данных. Во время каждой эпохи нейронная сеть обновляет свои веса для минимизации ошибки и улучшения производительности. Большое количество эпох зачастую помогает добиться лучших результатов, однако это не всегда так и чем их больше тем больше времени понадобиться на обучение. Так как взятая модель предобучена, для тестирования можно попробовать взять 15 эпох(для масштабов данной задачи это не слишком маленькое число эпох, однако порядка 25 было бы лучше, при этом всегда нужно смотреть практические результаты и делать поправки отталкиваясь от них)[16].

График отражающий результаты обучения можно увидеть на рисунке 4:

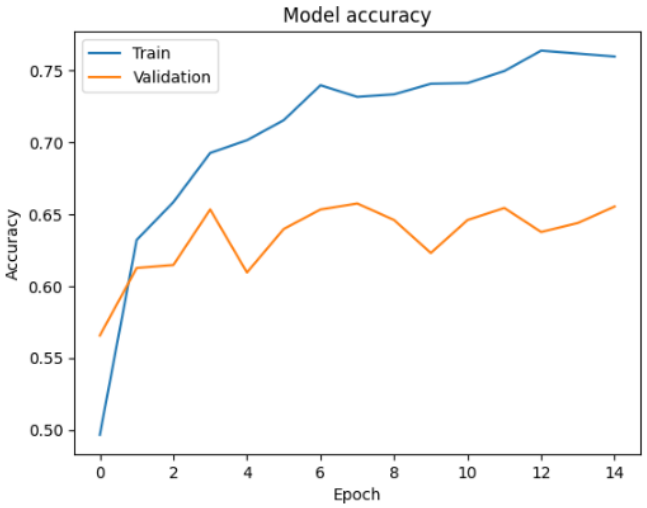


Рис.4 – результат обучения нейронной сети

Из данного графика видно что уже за 10 эпох точность модели достигла 75% для тренировочных данных и 65% для валидационных, также видно что прирост сильно замедлился уже после первых 6 эпох. Несмотря на то что график показывает признаки похожие на переобучение, здесь более вероятным фактором является проблема объекта классификации. Во многом тяжело добиться слишком высокой точности для импрессионистов в силу зачастую определенного числа работ которые сильно выходят за рамки закономерностей которые можно обнаружить с помощью нейронной сети. Импрессионизм, как направление в искусстве, характеризуется упором на передачу сути сцены, а не на детальный реализм. Художественные стили, особенно в области импрессионизма, весьма субъективны и могут сильно различаться даже в работах одного художника. Эта изменчивость затрудняет определение четких и последовательных признаков, которые можно надежно использовать для классификации. Различие между разными авторами-импрессионистами часто требует детального понимания их уникальных стилистических нюансов, которые могут быть тонкими и сложными для количественной оценки. Художественная интерпретация может внести в данные двусмысленность и шум. В разных работах одного и того же художника могут наблюдаться вариации из-за меняющихся влияний, личного развития или экспериментальных этапов, что усложняет задачу классификации[17,18,19].

Далее Обученную модель можно протестировать на тестовом наборе данных.

На рисунке 5 можно наблюдать матрицу построенную по результатам работы модели:

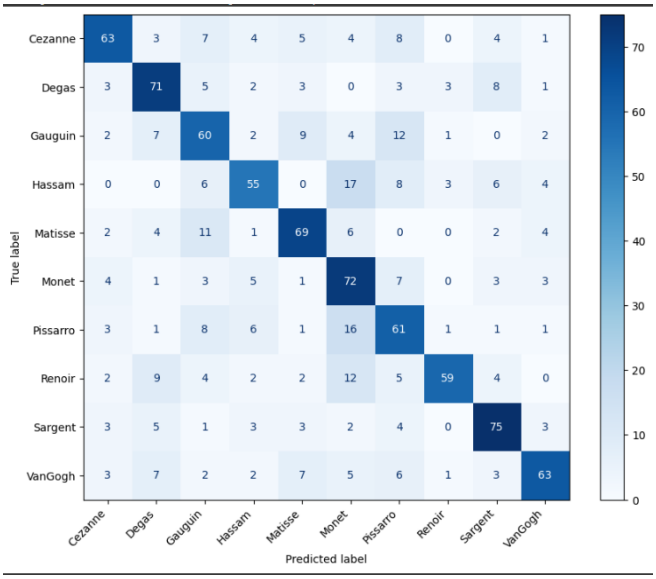


Рис.5 – матрица результатов

Если средняя точность для разных классов составляет около 70% по диагонали матрицы ошибок, это говорит о том, что модель достаточно хорошо справляется с правильной идентификацией экземпляров каждого класса.

Для достаточно сложной задачи классификации различных импрессионистов, точность равная 70% подтверждает целесообразность использования cnn для классификации картин[20].

V. Заключение.

В результате исследования была рассмотрена модель сверточной нейронной сети из библиотеки keras и адаптирована для классификации художников импрессионистов. Оценены возможности современных сверточных нейронных сетей в подобных задачах.

Для тестирования работоспособности классификатора использовался обширный набор данных состоящий из большей части существующих картин 10 художников импрессионистов. Полученные результаты наглядно продемонстрировали наличие ряда сложностей в сфере классификации картин по параметру автора, тем не менее продемонстрировав достаточно хорошие результаты.

Анализируя проделанную работу и все вышесказанное можно сделать вывод что нейронная сеть классификатор картин – перспективная разработка. Несмотря на некоторое количество уже разработанных инструментов в данной сфере, они все еще не достигли своего полного потенциала чтобы широко использоваться для идентификации подделок. Подобные продукты уже сейчас способны генерировать изображения и выполнять классификацию с достаточно хорошей точностью, однако до полной автоматизации процессов классификации объектов искусства и картин в частности не хватает еще нескольких шагов.

Литература.

[1] Аггарвал, Ч. Нейронные сети и глубокое обучение : учебный курс.– М. : Диалектика, 2020. – 744 с. : ил. – ISBN 978-5-907203-01-3.

[2] Ян Эрим Солек. Программирование компьютерного зрения на языке Python / пер. с англ. Слинкн А. А. - М.: ДМК Пресс, 2016 - 312 с.: ил.

[3] Fu, Feifei & Lv, Jiancheng & Tang, Chenwei & Li, Mao. (2020). Multi-style Chinese art painting generation of flowers. IET Image Processing. 15. 10.1049/ipr2.12059.

[4] Erkan, Uğur & Toktas, Abdurrahim & Enginoğlu, Serdar & Akbacak, Enver & Thanh, Dang. (2021). An Image Encryption Scheme Based on Chaotic Logarithmic Map and Key Generation using Deep CNN. 10.21203/rs.3.rs-440962/v1.

[5] Sunardi, Sunardi & Yudhana, Anton & WindraPutri, Anggi. (2022). Mass Classification of Breast Cancer Using CNN and Faster R-CNN Model Comparison. Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control. 10.22219/kinetik.v7i3.1462.

[6] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[7] Ali, Bushra & Sadekov, Rinat & Tsodokova, V.. (2023). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation. 13. 241-252. 10.1134/S2075108722040022.

[8] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[9] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[10] Solodov1, S.V., Mamai1, I.B. and Pronichkin2, S.V. (2022) IOPscience, IOP Conference Series: Earth and Environmental Science. Available at: https://iopscience.iop.org/article/10.1088/1755-1315/981/2/022007

[11] Arlazarov, V & Arlazarov, Vladimir & Bulatov, Konstantin & Chernov, Timofey & Nikolaev, Dmitry & Полевой, Дмитрий & Sheshkus, Alexander & Skoryukina, Natalya & Slavin, Oleg & Usilin, S. (2022). Mobile ID Document Recognition-Coarse-to-Fine Approach. Pattern Recognition and Image Analysis. 32. 89-108. 10.1134/S1054661822010023.

[12] Yu, Yue & Li, Ding & Li, Benyuan & Li, Nengli. (2023). Multi-style image generation based on semantic image. The Visual Computer. 1-16. 10.1007/s00371-023-03042-2.

[13] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov and L. -C. Chen, "MobileNetV2: Inverted Residuals and Linear Bottlenecks," 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 2018, pp. 4510-4520, doi: 10.1109/CVPR.2018.00474.

[14] Джулли, Пал: Библиотека Keras - инструмент глубокого обучения

[15] A. G. Howard et al. “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Application.,” Howard, Andrew G., et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications." arXiv preprint arXiv:1704.04861, 2017, pp. 1-9.

/ пер. с англ. А. А. Слинкин.- ДМК Пресс, 2017. – 249 с.

[16] Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. – СПб. : Питер, 2018. – 480 с. : ил. – ISBN 978-5-496-02536-2.

[17] T. Lin, M. Maire, S. J. Belongie, Hays et. al. “Microsoft COCO:

Common Objects in Context. European Conference on Computer Vision", Computer Vision (ECCV2014), 2014, vol. 8693, pp. 740-755.

[18] C. Szegedy, V. Vanhoucke, S. Ioffe et. al. "Rethinking the Inception Architecture for Computer Vision", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 2818-2826.

[19] Chen B. Classification of artistic styles of chinese art paintings based on the CNN model [J]. Computational Intelligence and Neuroscience, 2022, 2022: 1-7.

[20] K. Simonyan, A. Zisserman. "Very Deep Convolutional Networks for Large-Scale Image Recognition", The 3rd International Conference on Learning Representations (ICLR2015), 2014, pp. 1-14.

Исследование возможности классификации человеческих действий

И. Б. Алексеев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2311242@edu.misis.ru

П. Е. Злакоманов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2301834@edu.misis.ru

***Аннотация*— исследование направлено на понимание нейронными сетями человеческого поведения и присвоение класса каждому действию. Распознавание человеческих действий имеет широкий спектр применения и поэтому привлекает все большее внимание в области компьютерного зрения. Действия человека могут быть представлены с использованием различных модальностей данных, таких как RGB, скелет, глубина, инфракрасный порт, облако точек, поток событий, звук, ускорение, радар и сигнал Wi-Fi, которые кодируют различные источники полезной, но отдельной информации и имеют различные преимущества в зависимости от сценария и применения. В работе строятся модели классификации изображений с использованием CNN, которые классифицируют класс деятельности, выполняемый человеком на датасете с Kaggle и проверяется работа на реальных данных.**

Ключевые слова — Компьютерное зрение, Детекция человеческих действий, Распознавание человеческих действий, CNN, ResNet50, Xception, DenseNet169.

I. Введение

Нейронные сети нашли свое применение в разных отраслях. Например, в решениях проблем прогнозирования поведения транспортных средств на месте дорожного движения на основе анализа изображений. Для обнаружения объектов и оценки их местоположения используется 3D-детектор глубокой нейронной сети (DNN), где кинематическая модель велосипеда рассматривается как модель поведения, при которой каждое транспортное средство обрабатывается отдельно, без учета влияния других участников дорожного движения [1]. Также, в статье «Оценка точности трамвайной системы позиционирования в условиях высотного строительства с использованием данных визуальных геоинформационных систем» описан подход к оценке точности локализации трамвая, движущегося в городской среде, где трамвай должен быть локализован с точностью подметки. Поскольку информация GPS в городской среде не обеспечивает такого уровня точности, предлагается решение, основанное на использовании информации о системе зрения. Используя ключевые точки, можно оценить движение объекта между изображениями, полученными в разных проходах, в то время как это движение также можно рассчитать с помощью данных бортовой навигационной системы. Сопоставление этих перемещений позволяет оценить точность навигационной системы на борту [2]. В алгоритме автоматической посадки БПЛА с использованием компьютерного зрения рассматриваются алгоритмы систем зрения для автоматической посадки беспилотных летательных аппаратов (БПЛА). Представлены основные алгоритмы поиска вертолетной площадки и адаптации системы управления БПЛА к автономной посадке, рассмотрены алгоритмы решения навигационных задач, построения блоковых диаграмм автоматического управления. Разработанные алгоритмы позволяют реализовать автоматическую систему посадки для БПЛА с использованием систем технического зрения, с использованием различных параметров камеры, алгоритмов, позволяющих проводить исследования, в рамках автономной навигации беспилотных летательных аппаратов. В статье рассматривается разработанный алгоритм, позволяющий выделить определенные характерные точки для визуальной навигации. Кроме того, разработана система поиска характерных точек, позволяющая осуществлять автоматическую посадку БЛА. Эти алгоритмы могут быть полезны для автономных систем посадочных БЛА и для отслеживания траектории системы [3]. Аналогичная статья «Недорогая навигационная система для БПЛА» рассматривает малогабаритную навигационную систему NV-micro компании Integral Ltd, где представлены конструкция навигационной системы, алгоритм и результаты тестирования навигационной системы на световоммоторный самолет, демонстрирующий точность предлагаемой навигационной системы [4]. Виртуальное разворачивание или разворачивание, цифровое разворачивание, выравнивание или разворачивание - все эти термины используются для описания процесса выпрямления поверхности томографически реконструированного цифрового объекта. Цифровое выравнивание применяется в оптическом распознавании текста. В статье "От томографической реконструкции до автоматического распознавания текста: следующая задача для искусственного интеллекта" представлен открытый и кумулятивный набор данных CT-OPC-2022, который служит эталоном для реконструированных систем цифрового сплющивания и распознавания объектов [5].

Распознавание активности человека (HAR - Human Activity Recognition) - это известная исследовательская тема, которые используют камеры и микрофоны для записи движений тела: они не вмешиваются в личную жизнь пользователей, поскольку не включают видеозаписи в частных и домашних контекстах, менее чувствительны к окружающему шуму, дешевы и эффективны с точки зрения энергопотребления. Более того, широкое распространение встроенных сенсоров в смартфонах делает эти устройства повсеместными.

Одной из основных проблем в сенсорных методах HAR является представление информации. Традиционные методы классификации основаны на признаках, которые созданы и извлечены из кинетических сигналов. Однако эти признаки выбираются в основном на эвристической основе, в соответствии с поставленной задачей. Ча-

сто процесс извлечения признаков требует глубоких знаний в области применения или человеческого опыта и все же это приводит только к поверхностным признакам. Типичные методы HAR плохо масштабируются для сложных паттернов движения и в большинстве случаев не показывают хороших результатов на динамических данных, то есть данных, полученных из непрерывных потоков.

В этой работе мы применяем сверточные нейронные сети ResNet50, Xception, DenseNet169, которые имеют разные архитектуры, для исследования возможностей классификации HAR и сравниваем результаты.

Мы классифицируем действия человека, используя различные базовые модели с трансферным обучением, что требует средних вычислительных мощностей. В настоящее время в свободном доступе есть много данных для обучения.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались открытый набор данных, и собранные авторами. Рассмотрим используемый открытый набор.

A. *Kaggle HAR*

Мы взяли набор данных с Kaggle, который содержит:

- 15 различных классов человеческих действий.
- Около 12 тысяч помеченных изображений, включая изображения для валидации.

Каждое изображение относится только к одной категории человеческой активности и сохранено в отдельных папках для каждого помеченного класса.

На рисунке 1 представлены распределение классов активности:

Рисунок 1. Распределение классов активности

Рисунок 2. Примеры изображений

III. Нейросетевые архитектуры

A. *DenseNet169*

DenseNet представляет собой инновационную нейронную сеть, основанную на концепции skip connection. Структура DenseNet начинается с входного сверточного слоя, за которым следует блок DenseBlock. После этого принцип повторяется. Входные карты активации передаются каждому слою в блоке, обеспечивая плотное соединение информации.

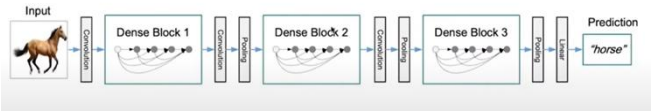


Рисунок 3. Структура DenseNet

Каждый блок DenseNet функционирует следующим образом: первый сверточный слой выдает карты активации, которые передаются следующему слою. Затем второй слой получает карты активации от обоих предыдущих слоев и выдает увеличенное количество карт. Процесс повторяется, увеличивая количество передаваемых карт активации с каждым последующим слоем.

Преимуществом DenseNet является высокий градиентный поток (strong gradient flow), что содействует борьбе с затуханием градиентов. Это позволяет создавать глубокие сети, например, DenseNet-264.

Кроме того, благодаря особенностям передачи информации между слоями, DenseNet эффективна в обучении, даже на небольших наборах данных.

Так как каждый сверточный слой внутри блока учитывает информацию из всех предыдущих слоев, сеть способна выделять разнообразные фичи. Нижние слои принимают во внимание более простые паттерны из верхних слоев, что может быть полезно для детекции низкоуровневых паттернов. Это делает DenseNet более эффективной на малых наборах данных.

B. ResNet

ResNet является сверточной нейронной сетью, используем ту, что содержит в себе 50 слоёв. На рисунке 1 представлена общая архитектура нейронной сети ResNet50.

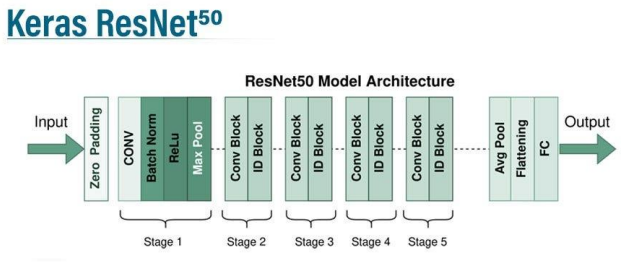


Рисунок 4. Архитектура ResNet50

После первого слоя и пулинга начинаются ResNet блоки, представленные на Рисунке 4. ResNet Block — это блок внутри Skip Connection, состоящий из двух слоев сети. На Рисунке 5 представлен пример ResNet Block.

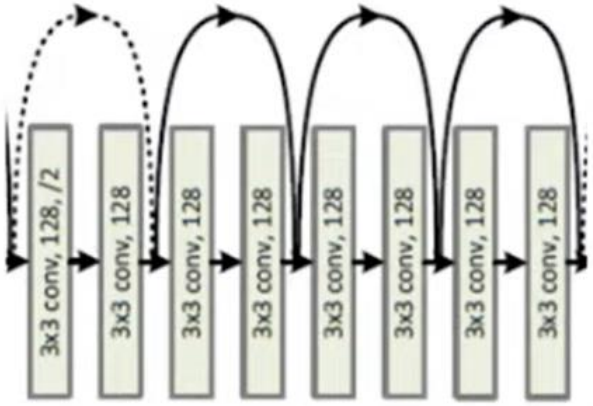


Рисунок 5. Пример ResNet Block

Пример ResNet Block включает 6 слоев, выдающих по 64 активации, затем 8 слоев с 128 активациями, и так далее.

Residual Blocks представляют ключевой элемент архитектуры ResNet, играя важную роль в обеспечении эффективности и глубины нейронной сети.

В каждом Residual Block сети ResNet присутствуют ровно два весовых слоя. Однако различия заключаются в том, как эти веса добавляются и взаимодействуют с Batch Normalization и ReLU.

ResNet одна из наиболее успешных архитектур в решении задач классификации изображений, поэтому решили использовать её тоже.

C. Xception

Inception схожа с концепцией ResNet. Вместо последовательных сверточных слоев в Inception используются несколько параллельных путей, основанных на сверточных слоях. Xception, представленная в 2017 году Франсуа Шолле и его командой, представляет собой эволюцию InceptionV3. Основная идея Xception заключается в полном пересмотре архитектуры Inception, заменив сверточные слои на глубокие разветвленные блоки глубокого разложения. Этот подход позволил модели эффективнее использовать параметры и повысить ее обобщающую способность. Содержит в себе 71 слой.

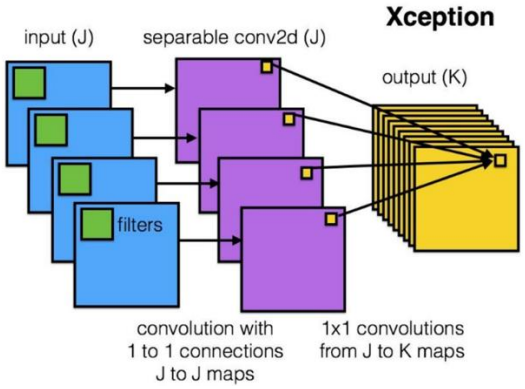


Рисунок 6. Пример Xception-модуль

На рисунке 6 и 7 представлен Xception-модуль и архитектура соответственно.

Данные сначала проходят через входной поток, затем через средний поток, который повторяется восемь раз, и, наконец, через выходной поток. Все слои Convolution и SeparableConvolution сопровождаются пакетной нормализацией.

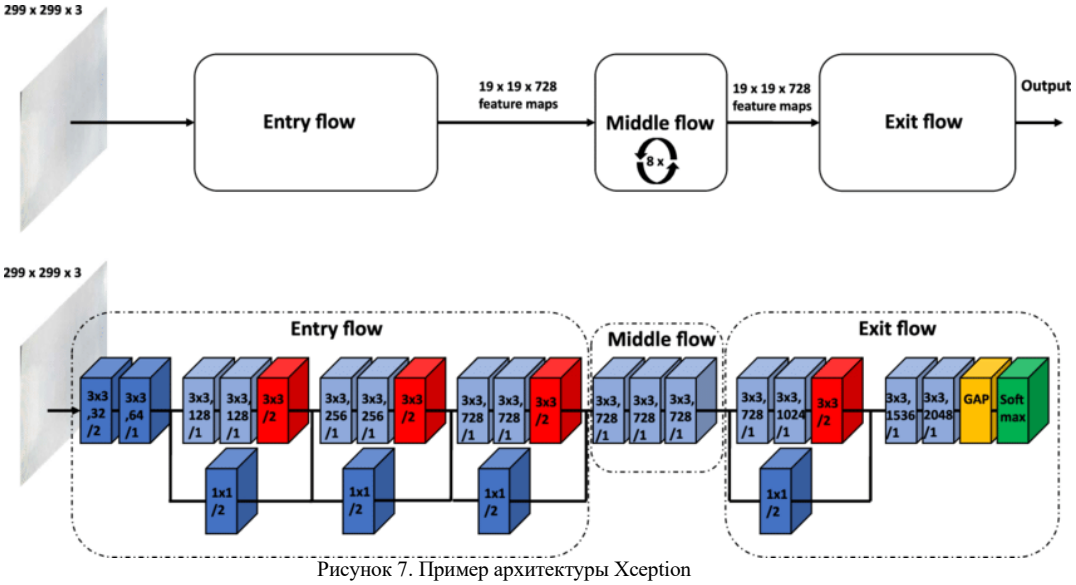


Рисунок 7. Пример архитектуры Xception

IV. Сравнение

Мы использовали такой подход, который включает в себя обучение 3 нейронных сетей с разными архитектурами и количеством слоёв на данных с Kaggle, а потом тестируем обученную модель на своём маленьком датасете, в котором преимущественно реальные данные.

Стоит отметить, что при обучении мы используем оптимизирующую функцию Adam. Основной идеей Adam является комбинация методов Momentum и RMSprop. Этот оптимизатор подстраивается под различные требования обучения, обеспечивая баланс между скоростью сходимости и адаптивностью к разным параметрам.

Adam поддерживает две переменные момента: первый момент (по аналогии с Momentum) и второй момент (по аналогии с RMSprop). Эти переменные вычисляются для каждого параметра модели.

Обновление первого момента (m): Отражает скорость изменения параметра

$$m_t = \beta_1 \times m_{t-1} + (1 - \beta_1) \times \nabla J_t$$

Обновление второго момента (v): Хранит информацию о квадрате градиента.

$$v_t = \beta_2 \times v_{t-1} + (1 - \beta_2) \times (\nabla J_t)^2$$

Коррекция смещения (bias correction): Учитывает начальные шаги оптимизации.

$$\hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$\hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

Обновление параметра (θ): Применяется для обновления весов модели.

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t}+\epsilon} \times \hat{m}_t$$

Где:

- ∇Jt - градиент функции потерь по параметру на шаге t
- β1 и β2 - коэффициенты затухания моментов
- η - шаг обучения
- ϵ - маленькое число для численной стабильности

Начинали мы с ResNet, т.к. она имеет наименьшее количество слоёв. Использовали тренированные веса Imagenet для каждой нейронной сети, каждая обучалась в 20 эпох.

ТАБЛИЦА I. Оценка точности и потерь на данных Kaggle после обучения

	ResNet50	**Xception**	**DenseNet169**
ValAccuracy	68%	74%	77%
TrainAccuracy	71%	77%	83%
TrainLoss	1.101	0.76	0.6
ValLoss	1.498	0.865	0.843
Recall	74%	74%	86%
Precision	87%	90%	88%

DenseNet169 демонстрирует лучшую производительность с точки зрения точности и потерь на валидационных данных, у неё самая высокая доля правильно классифицированных положительных случаев от общего числа предсказанных положительных случаев, т.к. у неё больше слоёв и она может увидеть больше признаков, в то время как ResNet50 проявляет наименьшую производительность среди рассматриваемых моделей, так же у неё наибольшие потери, что может указывать на то, что данные для неё слишком сложные. Xception находится между ними по производительности, обеспечивая умеренные результаты как по точности, так и по потерям, что является хорошим результатом, учитывая, что у неё 71 слой.

Чтобы проверить работу обученных моделей на реальных данных, мы собрали свой датасет, который содержит реальные изображения, сделали его разметку и взяли немного изображений с датасета Kaggle, примеры изображений на 8 рисунке.

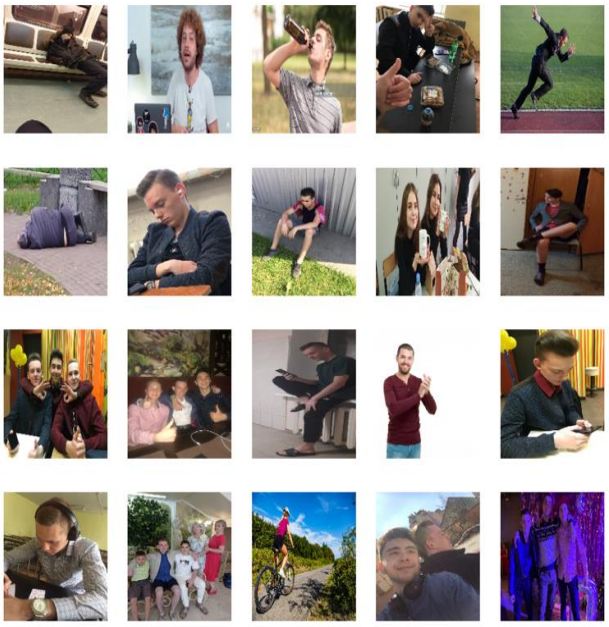


Рисунок 8. Примеры изображений

ТАБЛИЦА II. Результаты на реальных данных

	ResNet50	**Xception**	**DenseNet169**
Accuracy	13%	27%	27%

По результатам видно, что модели справились очень плохо, причём если изучить какие метки поставила сеть изображению, то видно, что реальные изображения он классифицирует, по большей части, ошибочно, а изображения с датасета Kaggle по большей части, правильно.

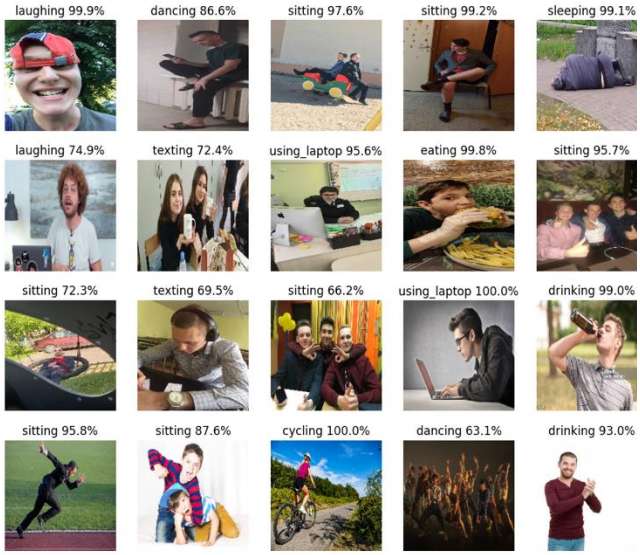


Рисунок 9. Примеры изображений с классификацией

В рисунке 9 представлены результаты классификации нейронной сети DenseNet169, у нашего датасета есть проблема с разметкой, потому что действие на одном изображении можно классифицировать по-разному, например, на одном изображении можно увидеть, что люди и сидят, и смеются, и обнимаются, если брать во внимание этот факт, то точность получается в районе 70%, касается DenseNet и Xception, модели более уверенно справляются со своими данными, потому что они более однозначные, т.е. простые. В итоге, можно сказать, что результат не такой уж и плохой, как кажется, но датасет с Kaggle слишком простой, чтобы использовать тренированные на нём модели где-то ещё.

V. Заключение

В заключение, модели были обучены на разнообразных данных, включая открытый набор Kaggle HAR, с последующим сравнением их результатов.

Результаты обучения на данных Kaggle показали, что DenseNet169 проявила лучшую производительность по сравнению с ResNet50 и Xception. Она продемонстрировала высокую точность классификации и более низкие потери как на обучающем, так и на валидационном наборах данных. С другой стороны, ResNet50 показала наименьшие результаты, что может быть связано с сложностью предоставленных данных.

Однако, при тестировании обученных моделей на собранном датасете реальных изображений, столкнулись с низкими результатами всех моделей. Вероятная причина заключается в неоднозначной разметке и сложности классификации действий на реальных изображениях.

Важно отметить, что полученные результаты подчеркивают важность корректной разметки данных и подготовки реальных датасетов для тестирования моделей. Несмотря на относительно высокую производительность на данных Kaggle, модели требуют доработки и настройки для более точного распознавания человеческих действий в реальных условиях.

Таким образом, дальнейшие исследования должны уделить внимание улучшению разметки данных, а также оптимизации параметров моделей для повышения их обобщающей способности на реальных изображениях.

Литература

[1] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," *2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[2] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[3] K. Dergachov, S. Bahinskii and I. Piavka, "The Algorithm of UAV Automatic Landing System Using Computer Vision," *2020 IEEE 11th International* Conference on Dependable Systems, Services and Technologies (DESSERT), Kyiv, Ukraine, 2020, pp. 247-252, doi: 10.1109/DESSERT50317.2020.9124998.

[4] D. B. Pazychev, K. S. Bakulev and R. N. Sadekov, "Low-Cost Navigation System for UAV," *2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS)*, Saint Petersburg, Russian Federation, 2023, pp. 1-6, doi: 10.23919/ICINS51816.2023.10168469.

[5] D. V. Polevoy, A. Ingacheva, "From tomographic reconstruction to automatic text recognition: the next frontier task for the artificial intelligence"

[6] Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., & Fei-Fei, L. (2014). Large-scale Video Classification with Convolutional Neural Networks. arXiv preprint arXiv:1412.0767. Available at: https://arxiv.org/abs/1412.0767

[7] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. arXiv preprint arXiv:1611.05431. Available at: https://arxiv.org/abs/1611.05431

[8] PyTorch Vision. (n.d.). DenseNet Implementation. Available at: https://github.com/pytorch/vision/blob/master/torchvision/models/densenet.py

[9] Xie, S., Girshick, R., Dollár, P., Tu, Z., & He, K. (2017). Aggregated Residual Transformations for Deep Neural Networks. arXiv preprint arXiv:1611.05431. Available at: https://arxiv.org/pdf/1611.05431.pdf

[10] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and Tell: A Neural Image Caption Generator. arXiv preprint arXiv:1411.4555. Available at: https://arxiv.org/pdf/1411.4555.pdf

[11] Дж. Смит и А. Джонсон, "Распознавание человеческих действий с использованием сверточных нейронных сетей", Журнал компьютерного зрения и обработки изображений, том 30, № 2, 2018, с. 45-62.

[12] Шапиро, Р. "Трансферное обучение в глубоком обучении: принципы и практика." https://arxiv.org/abs/1707.09725

[13] Шолле, Ф. "Xception: Deep Learning with Depthwise Separable Convolutions." https://arxiv.org/abs/1610.02357

[14] Huang, G. и др. "Densely Connected Convolutional Networks." https://arxiv.org/abs/1608.06993

[15] Kingma, D. P., и Ba, J. "Adam: A Method for Stochastic Optimization." https://arxiv.org/abs/1412.6980

[16] Методы оптимизации нейронных https://habr.com/ru/articles/318970/

[17] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 770-778) https://arxiv.org/abs/1512.03385

[18] Chollet, F. (2017). Xception: Deep Learning with Depthwise Separable Convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 1800-1807). https://arxiv.org/abs/1512.03385

[19] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. In Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR) (pp. 4700-4708) https://arxiv.org/abs/1608.06993

[20] Kingma, D. P., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. In Proceedings of the International Conference on Learning Representations (ICLR). https://arxiv.org/abs/1412.6980.

[21] https://www.researchgate.net/publication/367545589_A_Review_of_Navigation_Algorithms_for_Unmanned_Aerial_Vehicles_Based_on_Computer_Vision_Systems

[22] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

Распознавание людей на железнодорожной инфраструктуре

Д. И. Грищенко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2309680@edu.misis.ru

***Аннотация*— в настоящее время широкое распространение получают системы слежения, подключенные к системе распознавания. Подобные комплексы способны распознавать различные объекты, людей, животных, номера автомобилей. В зонах повышенной опасности, такой как железнодорожная инфраструктура, автоматические системы слежения должны распознавать наличие людей, а также то, насколько их экипировка соответствует стандартам нахождения в подобных зонах. Методы глубокого обучения показали высокую производительность и точность в данной области. В работе рассматривается решение с открытым исходным кодом и оценивается точность распознавания людей и наличие на них рабочей одежды по изображениям с камер видеонаблюдения на собственном наборе данных.**

Ключевые слова — Компьютерное зрение, Глубокое обучение, Сверточная нейронная сеть, Распознавание людей, Обеспечение безопасности на железнодорожной инфраструктуре, Kagley

I. Введение

Последнее столетие все большее внимания уделяется безопасности сотрудников при выполнении травмоопасных видов производства. Крупнейшие компании по всему миру тратят большие бюджеты на обеспечение безопасности труда своих работников. В законодательство государств вносят законы, регулирующие сферу охраны труда и вводящие ответственность для работодателей за их нарушение.

В компании ОАО «РЖД» выпускаются и регулярно актуализируются инструкции по охране труда для всех видов занятости сотрудников. В частности, одной из самых строгих является инструкция по охране труда для монтеров пути в связи с высоким уровнем опасности для жизни при выполнении работ. [1]

Одним из пунктов данной инструкции является обязательное ношение спецодежды. Автоматизированная система контроля на основе нейронных сетей должна помочь в контроле за исполнением требований по обеспечении безопасностей при проведении путевых работ. Должно контролироваться ношение сотрудниками светоотражающих жилетов и защитных касок при нахождении в зоне повышенной опасности.

Методы глубокого обучения показали высокую производительность и способность к обобщению во многих областях и типах задач, таких как классификация и обнаружение [2, 3, 4]. Сверточные нейронные сети хорошо себя показывают в подобных задачах, в том числе в распознавании людей. В работе рассматривается решение с открытым исходным кодом в области глубокого обучения для определения наличия на сотрудников путевых бригад рабочей одежды по изображениям 2D-камеры.

Подходы, основанные на обучении, особенно те, которые используют глубокое обучение, требуют больших объёмов аннотированных данных, что не всегда есть в наличии. В настоящей работе использовались собственные наборы данных. Однако следует также заметить, что подходы, основанные на обучении, требуют больших вычислительных мощностей.

II. Наборы данных

Для обучения и тестирования рассматриваемой в данной работе нейронной сети использовался набор данных, собранный автором.

A. *Набор данных для обучения*

Набор данных включает в себя изображения людей, находящихся на железнодорожных объектах. Данный собраны на сети железных дорог Российской Федерации. Изображения классифицированы по признаку наличия форменной спецодежды.

Всего представлено 4 класса:

- okey – присутствует светоотражающий жилет и каска;
- warning – отсутствует один из элементов спецодежды;
- bad – спецодежда отсутствует;
- unknown.

Всего в обучаемом наборе данных присутствует 2000 изображений всех классов.

На рисунке 1 представлены примеры изображений, находящихся в наборе данных для обучения.

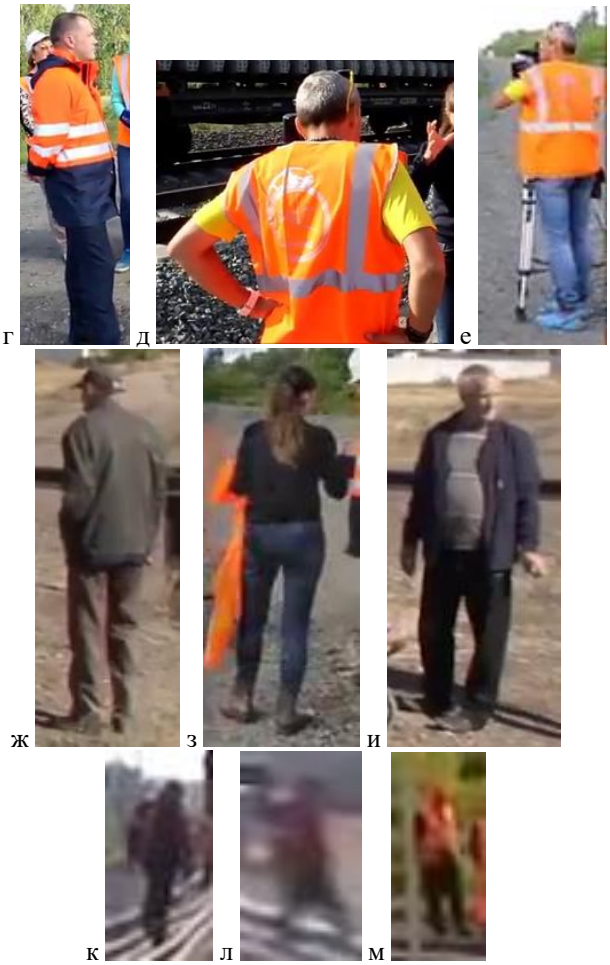


Рис. 1. Примеры изображений в наборе данных для обучения: а), б), в) okey, г), д), е) warning, ж), з), и) bad, к), л), м) unknown

B. Набор данных для проверки работоспособности

Второй набор данных используется для проверки точности рассматриваемой нейросети, полученной в результате обучения на первом наборе данных.

Данный набор не используется в самом процессе обучения, однако на нём происходит проверка полученной модели, выясняется, насколько точно модель способна различать входные данные. В этом наборе данных изображения собраны из таких источников, как фотографии и видеоматериалы из интернета, а также фотографии, сделанные автором статьи самостоятельно.

Данная выборка позволяет оценить, насколько обученная модель пригодна для использования на сети железных дорог с изображениями с реальных камер наблюдения.

В рассматриваемом наборе данных хранится около 1000 фотографий, разделенных на 4 класса по опасности нахождения человека на железнодорожной инфраструктуре.

Рис. 2. Примеры изображений в наборе данных для тестирования: а), б) okey, в), г) warning, е), ж) bad, з), и) unknown

III. Нейросетевые архитектуры

Свёрточная нейронная сеть в Keras

Для достижения поставленной задачи данном проекте используется свёрточная нейронная сеть в Keras. Сверточная нейронная сеть (Convolutional Neural Network или CNN) является распространенной архитектурой глубокого обучения, которая обрабатывает данные с применением операции свертки. [5] Свёрточные нейронные сети специализируются на обработке изображений и видео, хорошо улавливают локальный контекст, когда информация в пространстве непрерывна, то есть её носители находятся рядом. [6] Фреймворк Keras предоставляет простые и понятные методы для создания и обучения таких сетей [7].

Полная архитектура рассматриваемой нейросети представлена на рисунке 3.

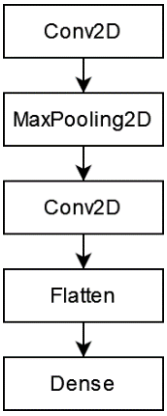


Рис. 3. Архитектура свёрточной нейронной сети

Первый сверточный слой Conv2D. Данный слой используется для извлечения признаков из изображений. Он применяет ядро свертки к входным данным, чтобы выделить различные шаблоны и структуры в изображениях. Этот слой создает ядро свертки, которое свертывается со входом слоя для получения тензора выходов. На слое используется 128 фильтров, матрица свертки имеет размер 3x3. На данном слое используется функция активации ReLU (rectified linear activation unit) — это кусочно-линейная функция, которая выводит входные данные без изменений, если они положительные, и ноль, если входные данные отрицательные. На вход поступает матрица размером 100 на 100 с палитрой пикселов 3 оттенка.

Далее идет слой пулинга MaxPooling2D. Данный слой уменьшает размерности данных. Он работает путем объединения информации из соседних участков входных данных и создания новых представлений с меньшим размером.

В качестве второго сверточного слоя используется Conv2D. Данный слой характеризуется наличием 256 фильтров, размер матрицы свертки составляет 3x3. Используется функция активации ReLU.

Далее следует слой конвертации Flatten, преобразующий формат изображения из двумерного массива в одномерный.

В качестве выходного слоя используется Dense, Он применяется для полносвязного соединения всех входных данных. На данном слое применяется функция активации Softmax. Функция Softmax возвращает вектор с вероятностями каждого возможного результата.

В рассматриваемой нейронной сети применяется оптимизационный алгоритм Adam (Adaptive Moment Estimation). [7] Определяется как метод эффективной стохастической оптимизации, который требует только градиентов первого порядка с небольшими требованиями к памяти. Он масштабирует скорость обучения, используя квадраты градиентов, аналогично RMSProp.

Метрикой выступает accuracy для вычисления частоты совпадения прогнозов с метками.

IV. Проведение тестирования

Для проведения тестирования необходимо обучить построенную модель. Основным параметром при обучении является количество эпох. Эпоха обозначает один проход через все обучающие примеры в заданном наборе данных. Во время одной эпохи нейронная сеть проходит через все входные данные и обновляет веса своих параметров, чтобы минимизировать ошибку и улучшить свою производительность. Чем больше эпох, тем больше времени потребуется для обучения сети, но при этом повышается шанс достижения лучшей производительности [4, 8].

Для проведения тестирования опытным путем было выявлено оптимальное количество эпох – 8. На данном количестве проходов нейронная сеть показывает оптимальные показатели по точности обучения на используемом наборе данных. Далее начинается спад показателей, что может говорить о переобучении модели.

Результаты обучения в течение 8 эпох представлены на рисунке 4.

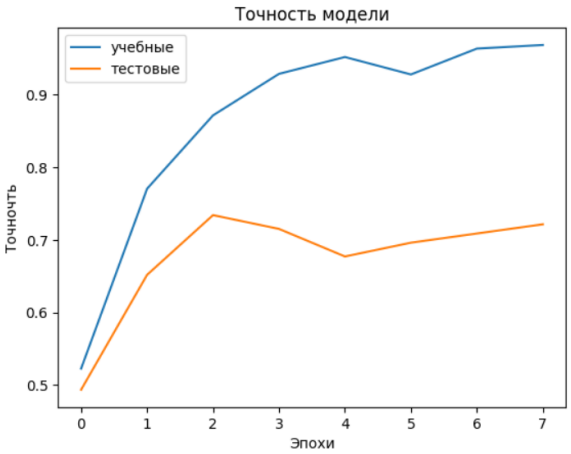
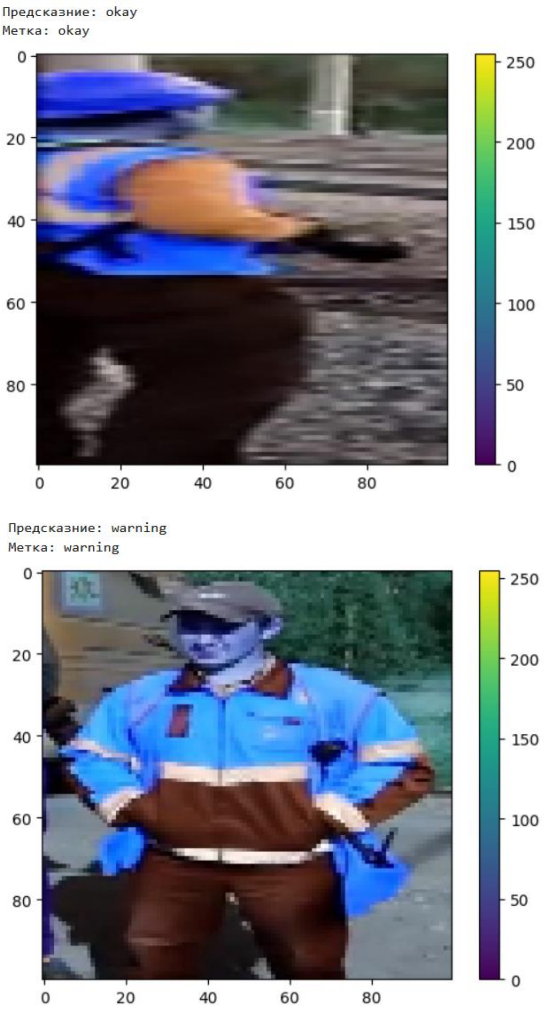


Рис. 4. Результат обучения сверточной нейронной сети

На данном графике можно видеть, что точность обучения на учебных данных достигла 98%. В то же время точность распознавания тестовых данных ниже и достигает 72%.

На рисунке 5 представлены примеры предсказания на тестовых данных.



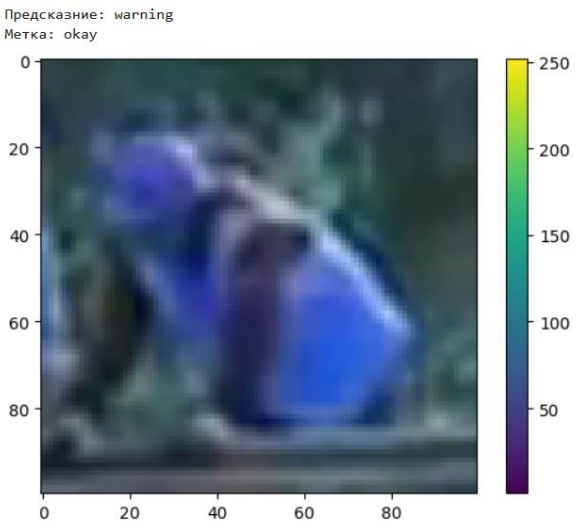


Рис. 5. Примеры предсказания на тестовых данных

На первом и втором примере видно, что предсказание совпало с меткой. На третьем примере нейронная сеть сделала ошибочное предсказание. Можно сделать вывод о том, что при обучении на имеющемся наборе данных модель с большей вероятностью справляется с более четкими изображениями, на которых люди расположены под прямым углом к камере.

Для повышения точности распознавания необходимо подготовить более обширный датасет с большим количеством изображений с нестандартными условиями, такими как непрямые углы съемки, плохие погодные условия, снижающие условия видимости.

В целом можно говорить о том, что рассматриваемая в рамках работы сверточная нейронная сеть справляется с поставленной задачей по обеспечению безопасности нахождения людей на железнодорожной инфраструктуре.

V. Заключение

В рамках проведенного исследования было рассмотрено готовое решение с открытым исходным кодом по распознаванию и классификации людей на железнодорожной инфраструктуре. Для обучения сети, построенной по принципу сверточной нейронной сети, была использована библиотека Keras.

Для обучения и тестирования было использовано два набора данных, которые были самостоятельно собраны авторами решения. Полученные в результате обучения модели результаты показали достаточно высокую эффективность при выполнении поставленной задачи по распознаванию людей. При должной доработке рассмотренная нейронная сеть может быть использована на сети железных дорог Российской Федерации для обеспечения охраны труда при проведении путевых работ.

Литература

[1] Инсутрукция по охране труда для монтера пути / ОАО «РЖД» - Москва, 2018.

[2] Аггарвал, Ч. Нейронные сети и глубокое обучение : учебный курс. – М. : Диалектика, 2020. – 744 с. : ил. – ISBN 978-5-907203-01-3.

[3] Ян Эрим Солек. Программирование компьютерного зрения на языке Python / пер. с англ. Слинкн А. А. - М.: ДМК Пресс, 2016 - 312 с.: ил.

[4] Николенко С., Кадурин А., Архангельская Е. Глубокое обучение. Погружение в мир нейронных сетей. – СПб. : Питер, 2018. – 480 с. : ил. – ISBN 978-5-496-02536-2.

[5] Джулли, Пал: Библиотека Keras - инструмент глубокого обучения / пер. с англ. А. А. Слинкин.- ДМК Пресс, 2017. – 249 с.

[6] Keras 3 API Documentation, available at: https://keras.io/api/ (Accessed: December 10, 2023)

[7] Полное руководство по оптимизации Адама, available at: https://skine.ru/articles/202703/ (Accessed: December 12, 2023)

[8] Рассел С., Норвиг П. Искусственный интеллект: современный подход, 2-е изд.. : Пер. с англ. - М. : Издательский дом ‘‘Вильямс’’, 2007. - 1408

Исследование возможности обнаружения текста произвольной формы

А. С. Корчевский
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2310312@edu.misis.ru

***Аннотация* — При распознавании текста произвольной формы точное определение границ текста является сложной и нетривиальной задачей. Существующие методы часто страдают от моделирования границ текста произвольной ориентации или сложной постобработки. В работе рассматриваются несколько решений с открытым исходным кодом и сравниваются возможности обнаружения текста на различных наборах данных.**

Ключевые слова — Компьютерное зрение, Обнаружение текста произвольной формы, Детекция текста, TextBPN-Plus-Plus, FAST, Total-Text, CTW1500.

I. Введение

Обнаружение текста является фундаментальной задачей компьютерного зрения с широким спектром практического применения, например, распознавание текста, поиск текста [1], обработка документов [2], мгновенный перевод, автопилотирование [3], [4], медицина [5]. Благодаря быстрому развитию обнаружения объектов с помощью сверточных нейронных сетей (CNN) и сегментации экземпляров, в задачах обнаружении текста был достигнут значительный прогресс и впечатляющая производительность для текстов правильной формы и правильного соотношения сторон. Обнаружение текста произвольной формы, как одна из самых сложных задач, вызывает постоянно растущий интерес как в научных, так и в промышленных кругах.

В отличие от обычного обнаружения объектов, где достаточно выделить текст с помощью рамки, обнаружение текста произвольной формы должно исследовать искривленные границы для каждого отдельного текстового региона. Методы, основанные на связных компонентах (connected components, CC) [6], [7] моделируют экземпляры текста с последовательными компонентами. Методы, основанные на сегментации [8], [9], моделируют экземпляры текста произвольной формы, прогнозируя маски на уровне пикселей, и определяют границы текста по краям масок. Как методы, основанные на CC, так и методы, основанные на сегментации, моделируют экземпляры текста с локальной точки зрения (локальные компоненты или пиксели) вместо непосредственного моделирования границ текста. Следовательно, они склонны пренебрегать глобальным геометрическим распределением общего расположения текста, что вызывает две основные проблемы: одна проблема заключается в том, что они чувствительны к шуму из-за однородной текстуры внутри текстовых областей; другая проблема заключается в том, что они будут полагаться на сложную и эвристическую постобработку для создания точных границ текста.

Недавно появилось множество решений на основе контуров [10], [11], [12], [13], [14] для непосредственного определения границ текстов произвольной формы и достижения многообещающей производительности. ABCNet [12] и FCENet [13] соответственно моделируют контуры экземпляра текста с помощью кривой Безье и кривой Фурье для эффективной регрессии замкнутых контуров. Некоторые другие методы [10], [11] используют нисходящие структуры обнаружения для регрессии ключевых точек на контурах текста с помощью операции ROI. Как утверждается в [14], эти методы воспринимают тексты со сложной геометрической разметкой только на одном этапе работы, что приводит к неточной локализации.

В данной статье мы рассмотрим несколько нейронных сетей [18], [19], выступающих в качестве решения проблемы обнаружения текста произвольной формы, а также сравним их характеристики на разных наборах данных.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались открытые наборы данных, которые будут разобраны ниже.

A. Total-Text

Набор данных Total-Text [15], выпущенный в 2017 году является первым, сочетающим в себе изображения с текстом 3 разных ориентаций (рисунок 1): горизонтальный (а), под наклоном (б) и изогнутый (в).

Рис. 1. Примеры каждого типа ориентации текста: а) горизонтальная, б) под наклоном, в) изогнутая

Особенность изогнутого текста заключается в том, что его нельзя сопоставить с прямой линией. Степень изогнутости текста в датасете Total-Text варьируется от небольшой до высшей (рисунок 2).

Рис. 2. Примеры разной степени изогнутости текста среди изображений Total-Text

Традиционно для обозначения обнаруженных объектов использовался прямоугольник, однако такой метод выделения не подходит для текста искривленной ориентации (рис. 3а). В качестве эталона, Total-Text предложили использовать многоугольник, состоящий из множества полигонов (рис. 3б).

а б

Рис. 3. Сравнение методов выделения объекта: а) традиционный (прямоугольник), б) новый (многоугольник)

B. *CTW1500*

Набор данных SCUT-CTW1500 [16], вышедший в том же 2017 году, включает в себя 1000 изображений для обучения и 500 изображений для тестирования, на которых содержится более 10 тысяч областей с текстом, и на каждом изображении как минимум один раз встречается изогнутый текст.

Здесь, как и в Total-Text, эталоном области является фигура, состоящая из нескольких полигонов. Пример изображений с выделением текста представлен на рисунке 4.

Рис. 4. Пример данных из набора CTW1500

III. Нейросетевые Архитектуры

A. *TextBPN-Plus-Plus*

В работе [17] решается задача обнаружения текста произвольной формы. Авторы предлагают использовать детектирующую нейронную сеть из трех модулей, которая в процессе обработки изображения совершает выделение границ текста по принципу “от грубого к точному”.

Метод состоит из трех модулей (рис. 5а):

- основной модуль определения признаков (архитектура отдельно изображена на рисунке 6);
- модуль предсказания границ (рис. 5б);
- модуль преобразования границ (рис. 5с).

После того, как основной модуль, обработав изображение, определяет признаки, модуль предсказания границ прогнозирует предварительную информацию для генерации “грубых” границ текста, что служит в свою очередь информацией для оптимизации модуля преобразования границ.

Модуль предсказания границ состоит из многослойных расширенных сверток, включая два сверточных слоя 3×3 с разными коэффициентами расширения и один сверточный слой 1×1, как показано на рисунке 5 (б).

В данном методе обнаружение текста произвольной формы происходит с помощью преобразования спрогнозированных грубых границ в точные границы текста. В частности, сеть обучается предсказывать смещения для каждой вершины, указывающей на границу текста, на основе грубых границ итеративным образом. Для каждого такого прогноза, представленного в виде замкнутого многоугольника, равномерно отбирается N контрольных точек (рис. 7), последовательность которых содержит не только контекст последовательности, но и топологический контекст (например, форму и пространственное распределение). Чтобы в полной мере использовать эти два контекста для уточнения грубых границ текста, авторы используют модуль преобразования границ, который эффективно изучает признаки и прогнозирует точные смещения для каждой вершины по направлению к границе текста.

Модуль преобразования границ использует структуру кодер-декодер, в которой кодер построен из многослойных преобразующих блоков с остаточным соединением, в то время как декодер представляет собой трехслойный персептрон (MLP) и сверку 1×1 с функцией активацией ReLU (рис. 5 (в)). Руководствуясь предварительной информацией, он постепенно уточняет прогнозы модуля предсказания границ с помощью итеративной деформации.

Кроме того, в этом методе используется инновационная функция потери энергии границ (boundary energy loss, BEL), которая вводит ограничение на минимизацию энергии и ограничение на монотонное уменьшение энергии для дальнейшей оптимизации и стабилизации процесса уточнения границ.

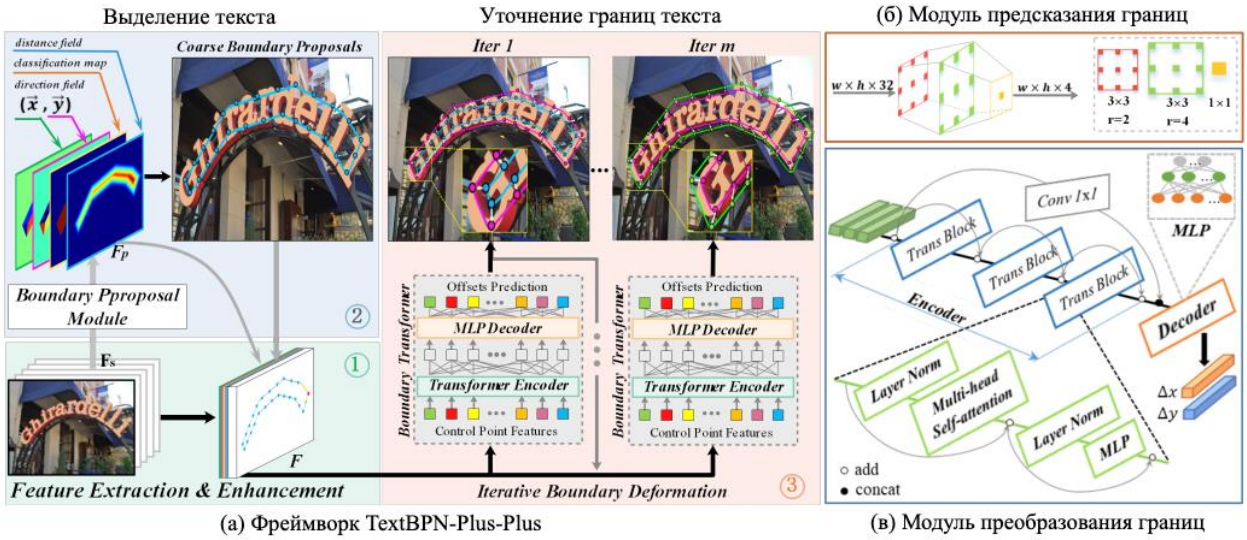


Рис. 5. (а) Полная модель фреймворка TextBPN-Plus-Plus. (б) Структура модуля предсказания границ. (в) Структура модуля преобразования границ

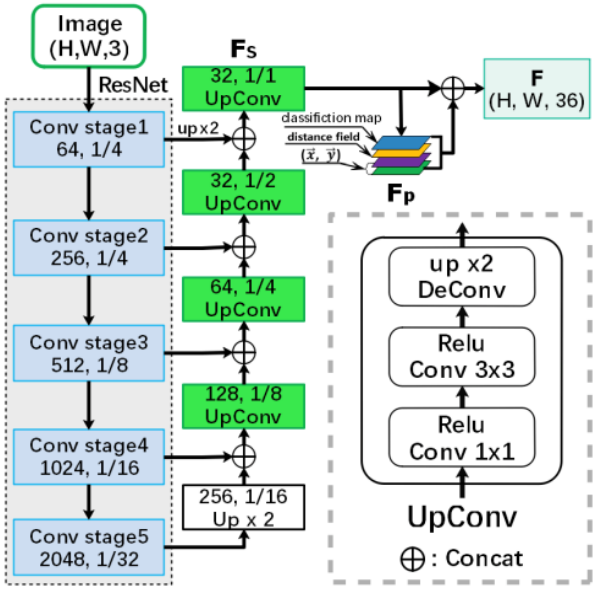


Рис. 6. Архитектура основного модуля определения признаков

Рис. 7. Иллюстрация уточнения границ с помощью модуля преобразования границ

B. *FAST*

Другой подход [18] заключается в использовании нейросетевой модели FAST (рис. 8), которая содержит в себе основную сеть с множеством поисковых блоков для улучшения архитектуры и модуля параллельной GPU-постобработки.

На стадии обнаружения текста, основной сети на вход подается изображение, которое проходит через четыре этапа выделения разномасштабных (1/4, 1/8, 1/16, 1/32) признаков. Затем их разрешение увеличивается и происходит их сложение в результирующую карту признаков F.

После этого карта признаков F проходит через двухслойную свертку, в результате которой получается прогнозируемый регион текста. Заключающим этапом выступает построение полных границ текста на GPU с помощью расширения предсказанного ранее региона.

Для упрощения пост-обработки используется инновационный подход, названный минималистичное представление ядра (minimalist kernel representation, MKR). Как показано на рисунке 9, MKR выражает строку текста в виде “вытравленного” текстового региона (т.е. текстовое ядро) с окрестными пикселями.

Для того, чтобы обучить сеть этому представлению, нужно сгенерировать метки для текстовых ядер и текстовых регионов. В частности, для данного текстового изображения метка текстовых областей может быть получена непосредственно путем заполнения ограничивающих рамок, которые обозначаются как G_{tex} (см. рис. 9(б)). Необходимо подчеркнуть, что G_{tex} – это двоичное изображение, и, если применять оператор удаления (erosion) с ядром s×s к G_{tex}, окрестные пиксели текстовых областей будут преобразованы в нетекстовые пиксели. Чтобы избежать потери самого текста, сохраняется по крайней мере минимальное текстовое ядро для каждой текстовой области. Этот результат берется в качестве метки для текстовых ядер и обозначается как G_{ker} (см. рис. 9(в)).

Во время тренировки используются функции потерь для оптимизации региона текста, прогнозируемого сетью, и границ текста, сгенерированных на этапе постобработки, соответственно. Во время поиска происходит расчет вознаграждения, исходя из точности полученных границ и скорости обнаружения, что затем используется для подкрепления сети. Процессы тренировки и поиска работают попеременно, после каждой итерации избыточные пути сокращаются, а архитектура оптимизируется.

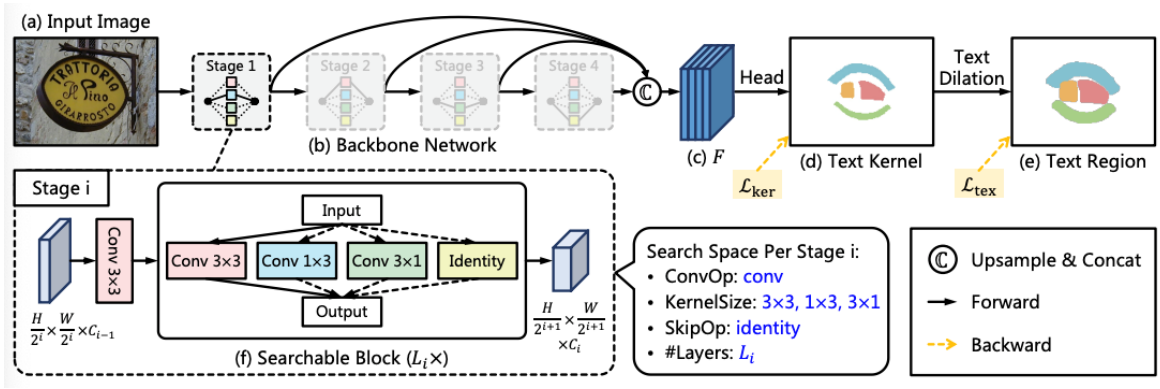


Рис. 8. Полная архитектура сети FAST

Рис. 9. Генерация меток MKR

IV. Сравнение

Сравним два описанных подхода. Для сравнения использовались нейросетевые модели, обученные на наборе данных Total-Text. Для сети TextBPN-Plus-Plus предобученная модель была взята с git-репозитория проекта [19], а для сети FAST было выполнено обучение на локальном ПК.

Тестирование моделей проводилось на датасетах Total-Text и CTW1500. В качестве метрик были выбраны precision (точность), recall (полнота), F-measure (баланс между точностью и полнотой), а также FPS (количество кадров в секунду, т.е. скорость модели).

Для чистоты эксперимента результаты работы модели TextBPN-Plus-Plus были приведены к формату результирующих данных модели FAST, а вычисление метрик проводилось с помощью одного и того же скрипта, взятого с git-репозитория проекта FAST [20]. Результаты тестирования приведены в таблице 1.

ТАБЛИЦА I. Результаты тестирования моделей

	Total-Text		CTW1500	
	TextBPN-Plus-Plus	FAST	TextBPN-Plus-Plus	FAST
Precision	0.9107	0.8730	0.3430	0.3868
Recall	0.8419	0.7509	0.5789	0.4997
F-measure	0.8750	0.8073	0.4308	0.4361
FPS	18.74	46.6	20.3	73.3

Как видно из таблицы модель FAST показывает лучшие результаты в скорости на обоих датасетах, полностью оправдывая свое название. Однако модель TextBPN-Plus-Plus обходит ее по всем остальным метрикам на наборе данных, который был использован для обучения.

Рассмотрим результаты оценки на датасете CTW1500. Увеличение скорости можно объяснить тем, что среднее разрешение изображений в этом наборе меньше, чем в наборе Total-Text. Значительное снижение оценок тоже закономерно: в этом датасете количество текстов искривленной ориентации, представляющих наибольшие трудности для обнаружения, больше, чем в датасете, использованном при обучении. Оценки моделей отличаются несильно (не считая FPS), тем не менее можно сказать, что модель FAST справилась лучше своего конкурента.

V. Заключение

Были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведенные подходы были сравнены на двух датасетах: Total-Text и CTW1500. По полученным данным очевидно, что, благодаря своей архитектуре, FAST имеет сильное преимущество в скорости перед альтернативным подходом. В точности обнаружения текста на датасете Total-Text, он все же уступает модели TextBPN-Plus-Plus, однако на наборе данных CTW1500, FAST превосходит конкурента по двум из трех метрик.

Литература

[1] Ilin, D., Novikov, D., Polevoy, D.V., & Nikolaev, D.P. (2018). Fast words boundaries localization in text fields for low quality document images. International Conference on Machine Vision.

[2] Arlazarov, V.L., Arlazarov, V.V., Bulatov, K.B., Chernov, T.S., Nikolaev, D.P., Polevoy, D., Sheshkus, A.V., Skoryukina, N.S., Slavin, O.A., & Usilin, S.A. (2022). Mobile ID Document Recognition–Coarse-to-Fine Approach. Pattern Recognition and Image Analysis, 32, 89-108.

[3] Ali, B., Sadekov, R.N., & Tsodokova, V.V. (2022). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation, 13, 241-252.

[4] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[5] Berdichevskaia A. Atypical lexical abbreviations identification in Russian medical texts //2022 12th International Conference on Pattern Recognition Systems (ICPRS). – IEEE, 2022. – С. 1-5.

[6] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, “Textsnake: A flexible representation for detecting text of arbitrary shapes,” in ECCV, 2018, pp. 19–35.

[7] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Liu, C. Yang, H. Wang, and X.-C. Yin, “Deep relational reasoning graph network for arbitrary shape text detection,” in CVPR, 2020, pp. 9699–9708.

[8] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, and C. Shen, “Efficient and accurate arbitrary-shaped text detection with pixel aggregation network,” in ICCV, 2019, pp. 8439–8448.

[9] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, “Real-time scene text detection with differentiable binarization,” in AAAI, 2020, pp. 11 474–11 481.

[10] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, “Arbitrary shape scene text detection with adaptive text region representation,” in CVPR, 2019, pp. 6449–6458.

[11] Y. Wang, H. Xie, Z.-J. Zha, M. Xing, Z. Fu, and Y. Zhang, “Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection,” in CVPR, 2020, pp. 11 753–11 762.

[12] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, “Abcnet: Realtime scene text spotting with adaptive bezier-curve network,” in CVPR, 2020, pp. 9806–9815.

[13] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, and W. Zhang, “Fourier contour embedding for arbitrary-shaped text detection,” in CVPR, 2021, pp. 3123–3131.

[14] P. Dai, S. Zhang, H. Zhang, and X. Cao, “Progressive contour regression for arbitrary-shape scene text detection,” in CVPR, 2021, pp. 7393–7402.

[15] Ch’ng, C.K., Chan, C.S.: Total-Text: a comprehensive dataset for scene text detection and recognition. In: IEEE 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), vol. 1, pp. 935–942 (2017).

[16] Liu, Y., Jin, L., Zhang, S., Luo, C., Zhang, S.: Curved scene text detection via transverse and longitudinal sequence connection. In: Pattern Recognition (2019).

[17] Zhang, S.-X.; Yang, C.; Zhu, X.; and Yin, X.-C. 2023. Arbitrary shape text detection via boundary transformer. IEEE Transactions on Multimedia.

[18] Chen, Z.; Wang, J.; Wang, W.; Chen, G.; Xie, E.; Luo, P.; and Lu, T. 2021. FAST: Faster Arbitrarily-Shaped Text Detector with Minimalist Kernel Representation.

[19] TextBPN-Plus-Plus official git-repository, available at https://github.com/GXYM/TextBPN-Plus-Plus/tree/main (Accessed: December 2, 2023).

[20] FAST official git-repository, available at https://github.com/czczup/FAST (Accessed: December 2, 2023).

Исследование возможности распознавания объектов на спутниковых снимках

Д. А. Рамзайцев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m1904484@edu.misis.ru

Д. С. Матяш
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m1910814@edu.misis.ru

***Аннотация* — обнаружение объектов на спутниковых снимках является важным направлением в сфере дистанционного зондирования земли. В данной работе проанализированы архитектуры нейронных сетей, используемых в задачах детекции, а также обучены 4 нейронные сети архитектуры YOLOv8 разных размеров на наборе данных, созданном на основе xView.**

Ключевые слова — Компьютерное зрение, Дистанционное зондирование земли, YOLO, xView, Обнаружение объектов

I. Введение

Анализ спутниковых снимков при помощи дистанционного зондирования земли играет важную роль в сферах городского планирования, борьбы со стихийными бедствиями и мониторингом окружающей среды [1].

Искусственные нейронные сети широко распространены в разных отраслях: в обнаружение машин и предсказании их передвижения [2], в оценке поз в робототехнике [3], управлении умным городом [4], навигации [5] и распознавании текста [6].

Хотя методы глубокого обучения показывают высокую эффективность и в задачах обнаружения объектов на изображениях, нахождение объектов на спутниковых снимках вызвано рядом сложностей [7]:

- различные размеры, разрешения изображений, а также высокое сходство между классами объектов;
- сложный фон, затрудняющий различие объекта от его окружения;
- ограниченность размеченных наборов данных и сложность создания таких наборов данных, вызванная большим количеством объектов на изображении и их наложении или пересечении.

В данной работе приведены результаты исследования по применению искусственных нейронных сетей в задачах распознавания объектов на спутниковых снимках. В частности, были обучены нейронные сети YOLOv8 четырех размеров.

II. Нейросетевые архитектуры

Распознавание объектов на изображениях заключается в определении их положения и категорий, к которым они принадлежат. Классические методы детекции объектов состоят из трех этапов [8]:

- поиск информативных областей;
- получение признаков;
- классификация.

Искусственные нейронные сети, в частности, сверточные архитектуры, имеют следующие преимущества:

- иерархическое представление признаков от пикселей к высокоуровневому описанию структуры изображения;
- комбинирование нескольких задач в одной модели, например поиск областей объектов на изображении и их классификация.

Выделяют два основных метода детектирования объектов [8]:

- метод на основе поиска регионов, содержащих объекты и последующей классификации объектов внутри найденных регионов. Задача детекции решается в два этапа (поиск регионов и классификация), и такие методы называют двухпроходными;
- метод, основанный на решении задачи регрессии и классификации. Поиск областей и определение их классов происходит в один этап, а сам метод называется однопроходным.

К двухпроходным методам относится архитектура R-CNN (Regions With CNNs), которая состоит из трех частей: CNN, bounding-box regressor (регрессия ограничивающих рамок) и SVM (классификатор на основе опорных векторов). Существуют и другие двухпроходные методы детекции с использованием нейронных сетей, но которые работают быстрее и точнее за счет некоторых улучшений. Среди них:

- Fast R-CNN: Обработка всех областей регионов одновременно и составление общей карты признаков для всех регионов. Совмещение трех моделей в одну;
- Faster R-CNN: Ускорение генерации регионов отдельной нейронной сетью RPN (Region Proposal Network) вместо алгоритма сегменации [9].

К однопроходным методам можно отнести следующие архитектуры нейронных сетей: YOLO (You Only Look Once), SSD (Single Shot MultiBox Detector), RetinaNet, EfficientDet. Также внутри одной архитектуры существуют разные модификации моделей, например, разное количество сверточных слоев, количество параметров.

Возможность распознавания объектов на спутниковых снимках с использованием нейронных сетей исследуется во множестве работ (SSD, Faster R-CNN и YOLOv3 [10], R-CNN и RetinaNet [11]). В некоторых задачах, когда объекты расположены плотно друг к другу и ограничивающие рамки сильно пересекаются, используют ориентированные (повёрнутые) ограничивающие рамки [12], [13], [14].

Среди архитектур нейронных сетей для детекции объектов наиболее популярна архитектура YOLO, поскольку она показывает наилучшее соотношение быстродействие/точность (начиная с YOLOv3), немного уступая по точности детекции (рисунок 1) и сильно опережая по скорости работы (рисунок 2).

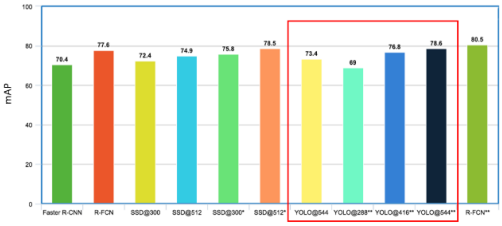


Рисунок 1 – Сравнение точности распознавания объектов с использованием различных архитектур нейронных сетей [15]

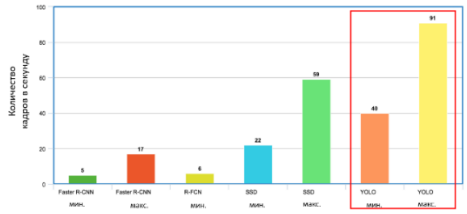


Рисунок 2 – Сравнение скорости работы различных нейронных сетей [15]

A. *Описание архитектуры YOLO*

Принцип работы архитектуры YOLO показан на рисунке 3. YOLO делит входное изображение на равные ячейки сеткой $S \times S$. Каждая ячейка сетки отвечает за предсказание объекта и предсказывает B ограничивающих рамок и соответствующие им оценки уверенности (вероятности).

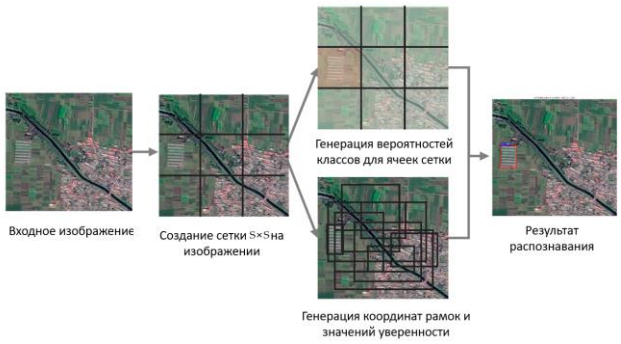


Рисунок 3 – Принцип работы архитектуры YOLO

После получения ограничивающих рамок производится подавление немаксимумов (non-maximum suppression, NMS [16]), которое позволяет избавиться от нерелевантных рамок. Это необходимо, потому что нейронная сеть часто находит несколько рамок, в которые попадает один и тот же объект (рисунок 4).

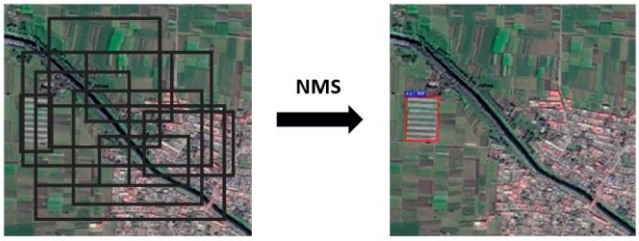


Рисунок 4 – Фильтрация нерелевантных ограничивающих рамок при помощи NMS

На вход сеть получает тензор $W \times H \times Ch$ являющийся изображением с шириной W, высотой H, и количеством каналов Ch. Выходной элемент для такой архитектуры представляет из себя тензор размера $S \times S \times (B \times 5 + n)$, где S – размер сетки (решетки), B – количество предикторов на одну ячейку сетки $S \times S$, n – количество классов.

Результирующий тензор состоит из векторов, содержащих $5 + n$ значений: $(P_{obj}\ b_x\ b_y\ b_w\ b_h\ P_1 \ldots P_n)$, где P_{obj} - уверенность (вероятность объекта в рамке), b_x, b_y - центры объекта относительно центра ячейки сетки, b_w, b_h - высота и ширина ограничивающей рамки относительно размеров всего изображения, $P_1 \ \ldots \ P_n$ – вероятности принадлежности объекта в рамке к одному из n классов.

В процессе развития архитектуры с YOLOv1 до YOLOv8 нейронная сеть получила ряд модификаций, существенно увеличивших ее точность и скорость работы [16]. Среди них:

- добавление слоев нормализации батчей;
- увеличение допустимого размера входного изображения;
- замена последних полносвязных слоев на сверточные (полностью сверточная сеть);
- обучение предсказыванию сдвигов якорных рамок фиксированной формы (может использоваться в задачах детекции, где объекты имеют одинаковую форму).
- обучение на изображениях разных размеров;
- улучшение основной (backbone) сети для извлечения признаков;
- поиск рамок на разных масштабах одновременно;
- модификация функции ошибки, минимизируемой в процессе обучения;
- оптимизация квантования для ускорения работы обученной нейросети.

Целевая функция, которая минимизируется в процессе обучения нейросети, описывает сдвиги центров рамок, размеров рамок относительно эталонных, а также вероятности нахождения объектов в них и их классы. Кроме того, минимизируется и DFL (Distribution Focal

Loss) для границ ограничивающих рамок [17], что увеличивает точность локализации объектов.

Схематично архитектура YOLOv8 изображена на рисунке 5. Сеть состоит из слоев двумерной свертки Conv2d, слоев уменьшения размерности MaxPool2d, слоев нормализации пакетов BatchNorm2d и слоев активации (SiLU), которые умножают входной сигнал на сигмоиду от входного сигнала. Эти слои, в свою очередь, сгруппированы по блокам, которые составляют основу сети.

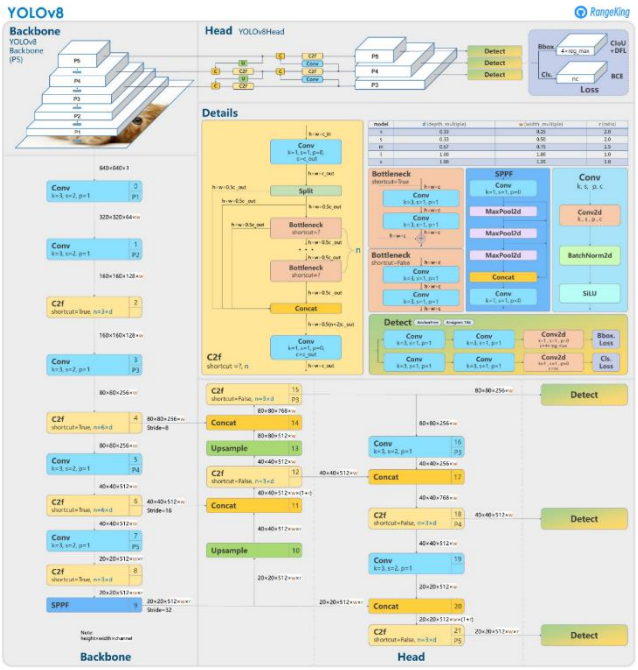


Рисунок 5 – Схематичное изображение архитектуры YOLOv8 [18]

Слои сети объединяются в блоки:

- Backbone – последовательность из блоков Conv и C2f, которые выполняют функции свертки и находят карты признаков, с пирамидой масштабов, которая используется для объединения признаков;
- Conv, C2f (coarse-to-fine) – являются составными частями сети Backbone;
- SPPF (spatial pyramid pooling fast) – пирамида масштабов, используемая для объединения признаков на разных масштабах;
- Upsample – слой повышения размерности карты признаков, используется для согласования входных размеров;
- Head – финальная последовательность слоев для формирования ограничивающих рамок и классов объектов при помощи блоков Detect;
- Detect – ряд слоев с использованием свертки с размером ядра 1, что позволяет уменьшать (или увеличивать) количество каналов. Подобные слои часто используются в полностью сверточных нейронных сетях в последних слоях сети.

B. Описание метрик

Для оценки качества распознавания объектов принято использовать 3 метрики: доля верно предсказанных моделью объектов (Precision), доля верно найденных моделью объектов среди всех объектов (Recall), а также mAP (mean Average Precision, среднее по Average Precision для каждого класса - площадь под кривой Precision-Recall).

Для оценки детекции объектов используется вспомогательная метрика IoU (intersection over union) для сравнения двух ограничивающих рамок, которая вычисляется как отношение площади пересечения к площади их объединения (рисунок 6).

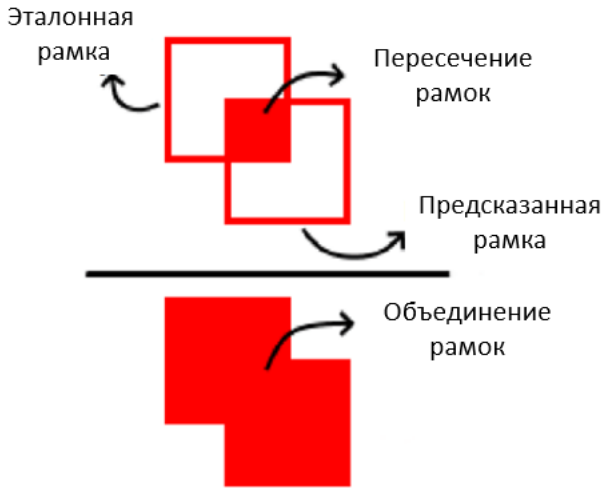


Рисунок 6 – Визуализация метрики IoU.

Метрика IoU лежит в диапазоне [0, 1] и чем больше ее значение, тем сильнее совпадают ограничивающие рамки. С ее помощью можно определить метрики Precision и Recall. Метрики задаются для двух множеств ограничивающих рамок $y = y_i$ и $\hat{y} = \hat{y}_i$. Каждая ограничивающая рамка содержит в себе индекс изображения, индекс класса, вероятность класса, координаты ограничивающей рамки. После детекции все рамки распределяются на 3 категории:

- предсказание считается True Positive когда метрика IoU для конкретной предсказанной рамки больше или равна некоторому пороговому значению;
- предсказание считается Flase Positive когда IoU меньше некоторого порогового значения, или является дубликатом (эталонная рамка уже была соотнесена с другой предсказанной рамкой);
- предсказание считается False Negative когда-либо на изображении с эталонной рамкой определенного класса не было найдено ни одного объекта (считается факт отсутствия предсказания).

Значения precision и recall находятся по следующим формулам:

$$precision = \frac{TP(c)}{TP(c) + FP(c)} \quad (1)$$

$$recall = \frac{TP(c)}{TP(c) + FN(c)} \quad (2)$$

где $TP(c)$ - количество предсказаний True Positive для класса c, где $FP(c)$ - количество предсказаний False Positive для класса c, где $FN(c)$ - количество предсказаний False Negative для класса c.

III. Набор данных

Основой обучающего набора данных стал набор данных xView, который содержит 60 классов и около 1 миллиона размеченных объектов. Из исходного набора данных были взяты 200 изображений, и каждое изображение было разбито на 9 более маленьких изображений. Часть классов были объединены в группы, другие классы не учитывались при обучении. В результате был получен набор данных, содержащий 1800 изображений и 3 класса: машина, самолет, лодка. Тестирование производилось на наборе данных из 180 изображений, полученных аналогичным образом. Пример полученного изображения показан на рисунке 7.

Рисунок 7 – Пример изображения в обучающей выборке

IV. Результаты и параметры обучения

Были обучены 4 модели YOLO размеров large, medium, small, nano. Обучение производилось на изображениях размера 1024x1024 и 30 эпохах с размером батча 4. В процессе обучения использовалась аугментация: Blur, MedianBlur, ToGray, CLAHE, Mosaic, Translate, Scale, Flip.

Полученные метрики на тестовом наборе данных приведены в Таблице 2. В качестве порога IoU использовалось значение 0.45, а в качестве порога Confiednce использовалось значение 0.25 для Precision и Recall. Для mAP использовался порог IoU, равный 0.5.

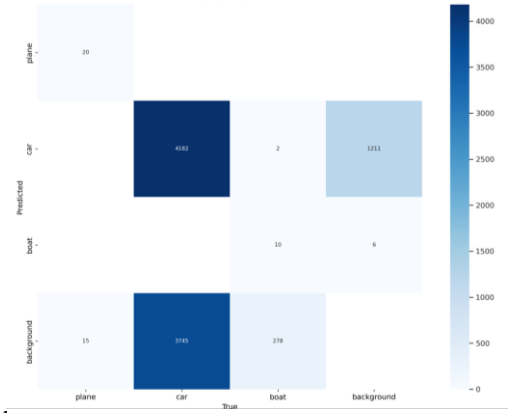
ТАБЛИЦА 2 Результаты обучения

	Precision	Recall	mAP
YOLOv8n	0.66928	0.41824	0.47535
YOLOv8s	0.66399c	0.46339	0.48132
YOLOv8m	0.626	0.49861	0.53936
YOLOv8l	0.73311	0.4683	0.49386

Более детальная статистика отображена для моделей YOLOv8n и YOLOv8l на матрице ошибок (рисунок 8а, рисунок 8б). В результате обучения большая модель в среднем показывает более высокую точность.

На рисунке 9 показаны графики зависимости метрик и ошибок от эпохи обучения. Графики строились тестовом наборе данных. Видно, что значения не достигают плато, метрики не начинают ухудшаться, а ошибки увеличиваться. Это говорит о том, что, с одной стороны, не произошло переобучение, с другой стороны, такое количество эпох недостаточно для обучения. Дальнейшее обучение моделей вызывает трудности, поскольку требует большого количества времени и вычислительных ресурсов.

а



б

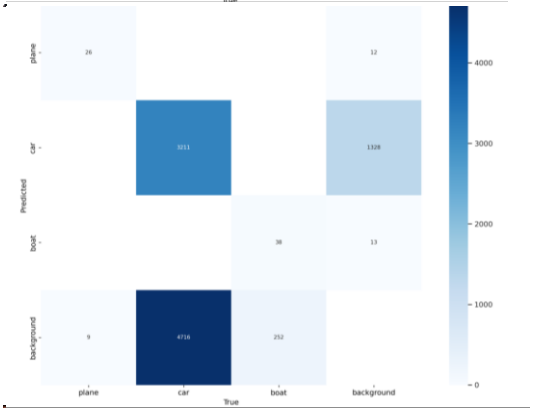


Рисунок 8 – Матрица ошибок YOLOv8l (а) и YOLOv8n (б)

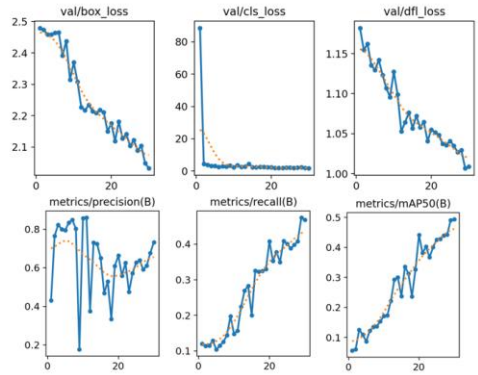


Рисунок 9 – Зависимость значений метрик и ошибок от эпох в процессе обучения модели YOLOv8l

Результат работы обученной YOLOv8l показан на рисунках 10-12.

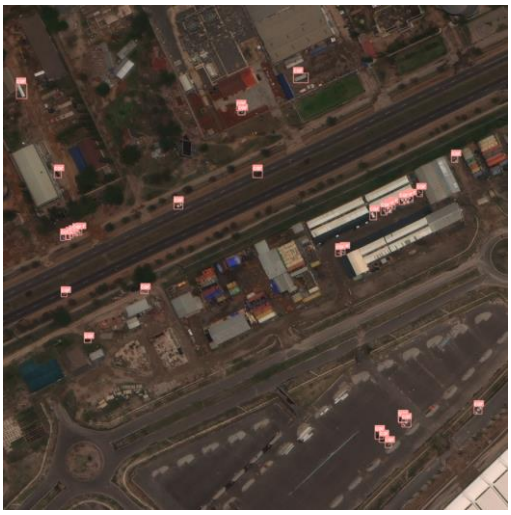


Рисунок 10 – Пример распознавания машин

Рисунок 11 – Пример распознавания лодок и машин

Рисунок 12 – Пример распознавания самолетов

V. Заключение

В данной работе были изучены методы обнаружения объектов на спутниковых снимках при помощи нейронных сетей. В частности, была описана, обучена и протестирована архитектура YOLOv8 на наборе данных сформированном на основе xView. Были сравнены 4 модели YOLOv8 размеров large, medium, small и nano. Также были описаны метрики, используемые в задачах обнаружения объектов.

Анализ метрик показал, что максимальная точность при обучении моделей не была достигнута, и требуется большее количество эпох для обучения. Тем не менее, полученные данные говорят о том, что чем больше модель, тем выше метрики. Несмотря на невысокие значения метрик, которые в свою очередь во многом обуславливаются дисбалансом классов и недостатком количества эпох, визуальные результаты детекции показывают, что архитектура YOLO позволяет находить небольшие объекты на спутниковых снимках и верно определять их класс.

Литература

[1] Bhuyan, K.; Van Westen, C.; Wang, J.; Meena, S.R. “Mapping and characterising buildings for flood exposure analysis using open-source data and artificial intelligence”, Nat. Hazards, vol 119. pp 1-31.

[2] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[3] S. H. Zabihifar, A. N. Semochkin, E. V. Seliverstova, and A. R. Efimov, “Unreal mask: one-shot multi-object class-based pose estimation for robotic manipulation using keypoints with a synthetic dataset,” Neural Computing and Applications, vol 33, Oct 2021, pp. 12283–12300, doi: 10.1007/s00521-020-05644-6.

[4] Y. S. Chernyshova, B. I. Savelyev, S. V. Solodov, S. V. Pronichkin, “Applying distributed ledger technologies in megacities to face anthropogenic burden challenges,” in IOP Conference Series: Earth and Environmental Science, 2022, vol. 1069, no. 1. doi:10.1088/1755-1315/1069/1/012028.

[5] B. Ali, R. N. Sadekov, V. V. Tsodokova, “A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems,” Gyroscopy and Navigation, vol. 30, pp. 87–105, 10.17285/0869-7035.00105.

[6] D. V. Polevoy, P. A. Kulagin, A. S. Ingacheva, Zh. V. Soldatova, M. V. Chukalina, D. P. Nikolaev, V. V. Arlazarov, "From tomographic reconstruction to automatic text recognition: the next frontier task for the artificial intelligence," Fifteenth International Conference on Machine Vision (ICMV 2022), 2023, vol. 12701. doi:10.1117/12.2680132.

[7] Adegun, Adekanmi Adeyinka and Fonou Dombeu, Jean Vincent and Viriri, Serestina and Odindi, John, "State-of-the-Art Deep Learning Methods for Objects Detection in Remote Sensing Satellite Images", Sensors, vol 23. 2023.

[8] Zhong-Qiu Zhao, Peng Zheng, Shou-Tao Xu, Xindong Wu, "Object Detection With Deep Learning: A Review", IEEE Transactions on Neural Networks and Learning Systems, vol. 99, pp 1-22, 2019.

[9] Pedro Felzenszwalb, Daniel Huttenlocher "Efficient Graph-Based Image Segmentation", International Journal of Computer Vision. vol. 59, pp 167-181, 2004.

[10] Li, M.; Zhang, Z.; Lei, L.; Wang, X.; Guo, X. "Agricultural greenhouses detection in high-resolution satellite images based on convolutional neural networks: Comparison of faster R-CNN, YOLO v3 and SSD". Sensors, vol. 2, 4938, 2020.

[11] Karim, S.; Zhang, Y.; Yin, S.; Bibi, I.; Brohi, A.A. "A brief review and challenges of object detection in optical remote sensing imagery", Multiagent Grid Syst., vol. 16, pp 227–243, 2020.

[12] J. Ding, N. Xue, Y. Long, G.-S. Xia, and Q. Lu, "Learning roi transformer for oriented object detection in aerial images," in CVPR, pp 2849–2858, 2019.

[13] X. Yang, J. Yang, J. Yan, Y. Zhang, T. Zhang, Z. Guo, X. Sun, and K. Fu, "Scrdet: Towards more robust detection for small, cluttered and rotated objects" ICCV, pp. 8232–8241, 2019.

[14] C. Li, C. Xu, Z. Cui, D. Wang, Z. Jie, T. Zhang, and J. Yang, "Learning object-wise semantic representation for detection in remote sensing imagery" CVPR Workshops, pp. 20–27, 2019

[15] "Two-stage vs One-stage Detectors" available at https://github.com/yehengchen/Object-Detection-and-Tracking/blob/master/Two-stage%20vs%20One-stage%20Detectors.md (Accessed: December 19, 2023).

[16] "A Comprehensive Review of YOLO: From YOLOv1 and Beyond" available at https://arxiv.org/pdf/2304.00501.pdf (Accessed December 19, 2023).

[17] "Generalized Focal Loss: Learning Qualified and Distributed Bounding Boxes for Dense Object Detection" available at https://www.researchgate.net/publication/342027416_Generalized_Focal_Loss_Learning_Qualified_and_Distributed_Bounding_Boxes_for_Dense_Object_Detection (Accessed December 19, 2023)

[18] "Ultralytics YOLOv8" available at https://github.com/ultralytics/ultralytics (Accessed December 19, 2023)

Преобразование текстовых запросов в 3D объекты

Н. И. Бугаков
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1900660@edu.misis.ru

В. О. Плотников
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1908601@edu.misis.ru

***Аннотация* — В данном исследовании проведено сравнение двух генеративных методов - Point·E и Shap·E - для формирования трехмерных облаков точек, ассоциированных с текстовыми описаниями. В отличие от прямого обучения единственной генеративной модели, использован трехэтапный процесс генерации. Модель GLIDE с миллиардом параметров применяется для формирования синтетических текстовых представлений, а условная модель диффузии используется для создания облаков точек низкого разрешения.**

Point·E представляет собой систему, создающую трехмерные облака точек с цветами RGB на основе сложных подсказок. В работе предложен метод для обработки данных, подробности которого описаны в разделе иллюстрации работы модели.

Shap·E, другой генеративный алгоритм, использует объединение различных подходов, кодировщик на основе Transformer для формирования неявных представлений, и диффузионную модель. Модели Shap·E прошли обучение на объемном наборе данных и показывают способность создавать разнообразные образцы, поддерживаемые текстовыми подсказками.

Сравнительный анализ результатов работы нейронных сетей Shap·E и Point·E при простых запросах показывает их схожую эффективность в создании объектов, однако выявляются различия в производительности при анализе большой выборки однотипных объектов. Результаты указывают на более высокую скорость сходимости и сравнимые или даже превосходящие результаты моделей Shap·E по сравнению с явной генеративной 3D-моделью Point·E при использовании аналогичной архитектуры и набора данных.

Ключевые слова — нейронные сети, 3D моделирование, текстовые запросы, изображения, виртуальная реальность, графический дизайн.

I. Введение

В последние десятилетия нейронные сети стали ключевым инструментом в области компьютерного зрения, позволяя решать широкий спектр задач, включая преобразование изображений 2D в трехмерные объекты. Этот процесс, известный как преобразование изображения 2D в 3D, представляет собой значительный вызов, требующий точности и высокой степени абстракции [1].

В данном обзоре мы исследуем различные методы и реализации использования нейронных сетей для выполнения преобразования изображений из двумерного пространства в трехмерные объекты. Разработанные алгоритмы включают в себя как классические подходы, основанные на глубоком обучении, так и современные техники, пытающиеся адаптировать уже существующие решения [2, 3].

В ходе обзора мы рассмотрим принципы работы различных методов, а также проведем сравнительный анализ их эффективности, точности и применимости в различных сценариях. Этот обзор предоставит полное представление о текущем состоянии исследований в области преобразования изображений 2D в 3D объекты с использованием нейронных сетей, а также выявит потенциальные направления для будущего развития этой области исследований [4].

В настоящее время значительный интерес исследователей привлекают неявные нейронные представления INR при кодировании трехмерных объектов. Эти представления стали особенно популярными для описания 3D-сцен, где INR сопоставляют 3D-координаты с различными характеристиками, такими как плотность и цвет. Важной особенностью является их независимость от разрешения, поскольку они могут быть запрошены в произвольных точках входных данных, в отличие от кодирования информации в фиксированной сетке или последовательности [5].

Исследования в области INR (Неявное Нейронное Представление) обрели значительное значение, так как они обладают свойствами сквозной дифференцируемости [6]. Это открывает широкие возможности для различных последующих приложений, включая передачу стиля и редактирование дифференцируемых форм. В данной работе мы сосредоточимся на рассмотрении двух основных типов INR для 3D-представления:

1. **Поле нейронного излучения (NeRF):** NeRF представляет собой INR, описывающее 3D-сцену в виде функции, которая сопоставляет координаты и направления просмотра с плотностями и цветами RGB. Этот метод позволяет отрисовывать изображения 3D-сцены из произвольных точек, запросив плотности и цвета вдоль лучей камеры. Он обучается создавать реалистичные изображения 3D-сцен, что делает его мощным инструментом для компьютерного зрения.
2. **DMTet и GET3D:** DMTet (Дифференцируемые тетраэдрические сетки) и его расширение GET3D (Генеративная занятость тетраэдрических трехмерных сеток) представляют трехмерные текстурированные сетки как функцию, отображающую координаты в цвета, расстояния со знаком и смещения вершин. Этот тип INR позволяет создавать трехмерные треугольные сетки дифферен-

цируемым образом, и результаты могут быть эффективно визуализированы с использованием дифференцируемых библиотек растеризации.

II. Наборы данных

Для выполнения наших экспериментов мы опираемся на предварительно обученные нейронные сети с общим набором данных базовых 3D-ресурсов, что обеспечивает более справедливое и сопоставимое сравнение с их методом. Также известно, что эта сеть была расширена и изменена по сравнению с оригинальным набором данных. Вот какие изменения были внесены:

1. Увеличено количество визуализаций: Для расчета облаков точек теперь визуализируется 60 видов каждого объекта вместо стандартных 20. Это решение было принято в ответ на наблюдение, что использование всего 20 изображений иногда приводило к появлению мелких трещин в предполагаемых изображениях облаков точек из-за слепых зон [7].
2. Увеличен размер облаков точек: Теперь увеличено количество точек в облаках до 16 тысяч вместо обычных 4 тысяч. Это изменение помогает более точно описывать геометрию объектов и улучшает качество реконструкции.
3. Упрощена постобработка для обучения кодировщика: При создании изображений для обучения кодировщика, было упрощено освещение и материалы. Все модели визуализируются с фиксированной конфигурацией освещения, поддерживающей только диффузное и окружающее затенение. Это упрощение улучшает соответствие настроек освещения с дифференцируемым рендерером, способствуя более стабильному обучению.

Для условной модели текста и соответствующей базовой линии Point·E используется расширенный набор данных базовых 3D-активов и текстовых подписей. Этот набор данных включает около 1 миллиона дополнительных 3D-ресурсов, собранных из высококачественных источников данных. Кроме того, было собрно 120 тысяч подписей от людей, размечающих высококачественные подмножества нашего набора данных. В процессе обучения моделей преобразования текста в 3D, случайным образом выбирается между ярлыками, предоставленными человеком, и оригинальными текстовыми подписями, если они оба доступны. Это обогащение данных способствует обучению более универсальных и точных моделей

III. Оптимизация точности и разнообразия: Метод гауссовой диффузии в создании диффузионных моделей облака точек

Метод основан на последних исследованиях моделей, использующих диффузию, предложенных Солом Дикштейном и коллегами (2015) [8] и Сонг & Эрмон, (2020) [9]. В работе применяется схема гауссовой диффузии, предложенная Хо и соавторами (2020) [10, 11].

Цель — извлекать образцы из распределения q(x0) с использованием нейросетевого приближения pθ(x0). В рамках гауссовой диффузии определяется зашумляющий процесс

$$q(x_t|x_{t-1}) := N(x_t; \sqrt{1-\beta_t}x_t - 1,\ \beta_t\ I)$$

для временных шагов t ∈ [0, T]. Этот процесс последовательно добавляет гауссовский шум к сигналу, причем величина шума зависит от графика шума βt. График шума выбирается так, чтобы на последнем временном шаге t = T выборка xT содержала минимум информации (почти выглядела как гауссовский шум).

Процесс зашумления можно представить напрямую, используя формулу

$$x_t = \sqrt{\alpha_t}\, x_0 + \sqrt{1-\alpha_t} \in$$

Для обучения диффузионной модели аппроксимируется q(xt−1|xt) при помощи нейронной сети pθ(xt−1|xt). Затем создается выборка, начиная с случайного гауссовского шума x_T, и процесс шумообразования обращается назад до достижения бесшумного состояния образца x_0.

Для обеспечения баланса между разнообразием выборки и точностью диффузионных моделей используются различные стратегии. Дхаривал и Никол (2021) [12] представляют руководство по классификатору, учитывающему шум, для искажения каждого шага выборки. Хо и Салиманс (2021) [13] предлагают руководство без использования классификаторов, где условная диффузионная модель p(xt−1|xt, y) обучается с использованием меток класса.

Диффузионные выборки можно рассматривать через призму дифференциальных уравнений, что позволяет использовать различные решатели SDE (стохастическое дифференциальное уравнение) и ODE (обыкновенное дифференциальное уравнение) для выборки из этих моделей Сонг и др., (2020) [14]. Используется решатель ОДУ второго порядка для обеспечения компромисса между качеством и эффективностью отбора проб Каррас и др., (2022) [15].

В общем, используется метод гауссовой диффузии для создания диффузионных моделей облака точек, стремясь найти оптимальный баланс между точностью и разнообразием в процессе обучения.

IV. Сравнение генеративных методов Point·E и Shap·E

Вместо прямого обучения единственной генеративной модели для формирования облаков точек, ассоциированных с текстом, выбирается иной метод. Процесс генерации подразделяется на три этапа. Инициируется создание синтетического вида, зависящего от текстовой подписи. Затем создается грубое облако точек (1024 точек), соответствующее сгенерированному синтетическому виду. В заключение формируется тонкое облако (4096 точек), соответствующее низкоразреженному облаку и синтетическому виду. Предполагается, что изображение содержит информацию из текста, не представляя явно облака точек в тексте.

Для формирования условных текстовых синтетических представлений применяется модель GLIDE с миллиардом параметров Никол и др., (2021) [16], тщательно настроенная на визуализированных 3D-моделях из набора данных. Для формирования облаков точек низкого

разрешения используется условная модель диффузии, инвариантная к перестановкам.

Для повышения дискретизации облаков точек с низким разрешением используется аналогичная (но меньшая) модель диффузии, дополнительно зависящая от облака точек низкого разрешения. Обучение моделей проводится на многомиллионном наборе данных 3D-изображений, с соответствующей метаинформацией, обрабатываемой в визуализированные виды, текстовые описания и трехмерные облака точек [17].

Point·E представляет собой систему для создания трехмерных облаков точек на основе сложных подсказок с соответствующими цветами RGB для каждой точки. Подробности по обработке данных описаны в разделе (рисунок 1) [18].

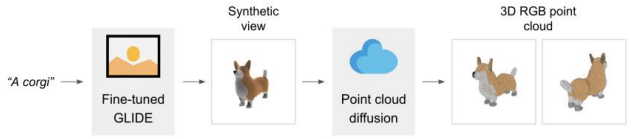


Рис. 1. Представление работы модели на самом высоком уровне Point·E

A. *Shap·E*

Объединение и масштабирование разнообразных подходов является ключевым этапом в разработке Shap·E, инновационного генеративного алгоритма, спроектированного для создания богатых и сложных трехмерных неявных представлений. Первым шагом в этом процессе является расширение методологии Чена и Ван [19] через обучение кодировщика на основе Transformer для формирования параметров Неявного Нейронного Представления (INR) для 3D-активов.

Далее, в соответствии с подходом Дупонт и др., проводится обучение диффузионной модели на выходах кодировщика. В отличие от предыдущих методов, предлагается концепция создания INR, объединяющих в себе элементы NeRF и сеток. Это открывает возможность визуализации моделей несколькими способами и легкость их интеграции в последующие 3D-приложения [20, 21, 22].

В процессе обучения на объемном наборе данных, включающем миллионы 3D-ресурсов, разработанные модели демонстрируют способность создавать разнообразные и узнаваемые образцы, которые поддерживаются текстовыми подсказками. По сравнению с недавно предложенной явной генеративной 3D-моделью Point·E, наши модели проявляют более высокую скорость сходимости и достигают сравнимых или даже превосходящих результатов при использовании аналогичной архитектуры модели, того же набора данных и механизмов обработки [7].

B. *Сравнение результатов работы нейронных сетей.*

Исследование результатов генерации нейронными сетями Shap·E и Point·E при простых запросах показало, что обе модели демонстрируют схожую эффективность в создании объектов (рисунок 2). На первый взгляд, при обработке общих запросов, таких как простая генерация объектов, обе модели успешно сохраняют высокий уровень детализации и качества. Однако, при анализе результатов на большой выборке однотипных объектов, таких как скейтборды, выявились различия в их производительности.

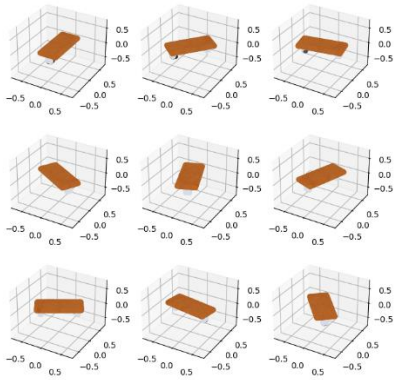


Рис. 2. Результат работы Point·E при запросе "skateboard"

Shap·E и Point·E обладают способностью генерировать общие объекты с высоким качеством, но при работе с многократно повторяющимися объектами, такими как "skateboard", Shap·E проявляет тенденцию к крайне искаженным формам (рисунок 3). Вероятно, это связано с особенностями обучения модели или ее чувствительностью к конкретному типу данных.

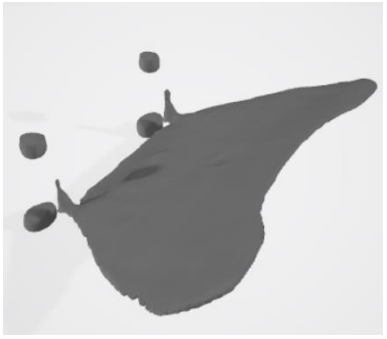


Рис. 3. Искаженный результат работы Shap·E при запросе "skateboard"

С другой стороны, объекты, созданные Shap·E, обладают более натуралистичной геометрией, что можно интерпретировать как более реалистичное воссоздание формы объекта (рисунок 4). Это свидетельствует о способности Shap·E сохранять естественные пропорции и формы при генерации объектов, что может быть важным при работе с реальными данными и визуализациями.

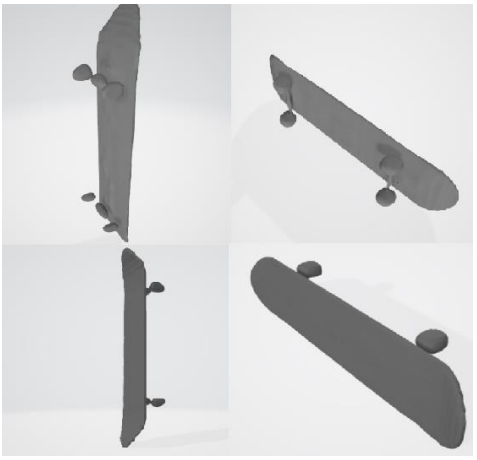


Рис. 4. Результат работы Shap·E при запросе "skateboard"

В отношении сложных запросов, таких как "A black man is standing on a chair", выявляется значительная разница в подходе Shap·E и Point·E. Shap·E демонстрирует тенденцию к буквальному воссозданию лексического значения фразы, добавляя к 3D-объекту стул. Это может привести к появлению артефактов и неестественных сценариев, не соответствующих ожидаемому смыслу запроса (рисунок 5).

Рис. 5. Результат работы Shap·E при запросе "A black man is standing on a chair"

С другой стороны, Point·E более адекватно интерпретирует сложные запросы, создавая изображение, которое соответствует смыслу запроса. Однако наблюдается игнорирование уточняющих характеристик в запросе, что может снизить точность соответствия в определенных случаях (рисунок 6).

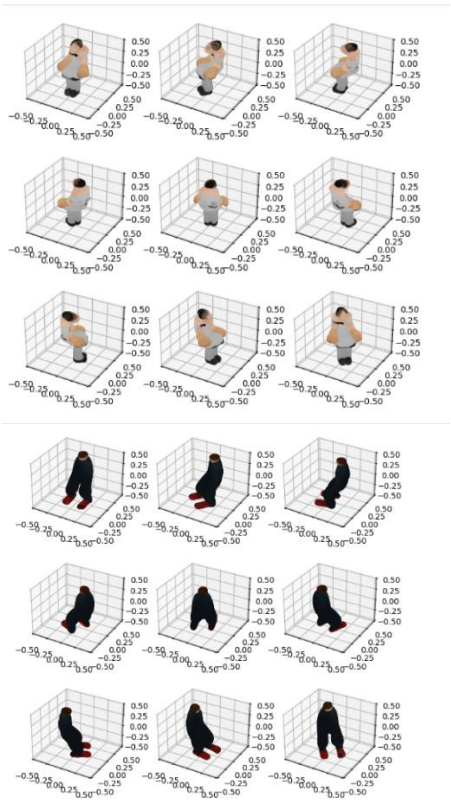


Рис. 6. Результат работы Point·E при запросе " A black man is standing on a chair"

C. Сравнение скорости работы нейронных сетей.

В начале работы с данными нейронными сетями возникла проблема долгой обработки запросов на видеокарте NVIDIA RTX 3050. Это привело к неэффективной работе и затруднило нормальное функционирование продукта. Для решения данной проблемы было принято решение внести изменения в библиотеку, чтобы обеспечить более эффективную работу с видеокартой, используя технологию CUDA. Эти изменения позволили продукту более эффективно использовать вычислительные ресурсы видеокарты RTX 3050 и справляться с задачами обработки данных гораздо быстрее.

ТАБЛИЦА I. Результаты Тестирования на Видеокарте NVIDIA RTX 3050

Модель	Среднее время обработки запроса (в секундах)	Минимальное время обработки запроса	Максимальное время обработки запроса
Shap·E	30	29	33
Point·E	41	40	42

Таблица представляет результаты тестирования нейронных сетей Shap·E и Point·E на видеокарте NVIDIA RTX 3050. Средние значения времени обработки запроса, а также минимальные и максимальные значения, отражают эффективность каждой модели в работе с данными запросами.

V. Заключение

Исходя из результатов скорости работы моделей на видеокарте NVIDIA RTX 3050, можно отметить, что Point·E проявляет более высокую эффективность в обработке запросов по сравнению с Shap·E. Среднее время обработки запроса для Shap·E составляет примерно 30 секунд, в то время как Point·E демонстрирует более быстрый результат, среднее время обработки приближается к 41 секунде.

В контексте сложных запросов, где требуется более реалистичное и адекватное понимание контекста, Point·E является более предпочтительным выбором. Эта модель способна сохранять форму человека, игнорируя вспомогательные объекты, и в целом обеспечивает более высокую скорость обработки. В отличие от этого, Shap·E проявляет тенденцию к прямому воссозданию лексического значения, что может привести к введению несоответствующих артефактов.

Таким образом, при выборе между Shap·E и Point·E рекомендуется учитывать требования конкретной задачи

и предпочтения пользователя, прислушиваясь к желаемому уровню скорости и адекватности восприятия текстовых запросов.

Литература

[1] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N. Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene," 2022 29th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388.

[2] N. S. Guzhva, B. Ali, K. S. Bakulev, R. N. Sadekov and A. V. Sholokhov, "Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems," 2023 30th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), Saint Petersburg, Russian Federation, 2023, pp. 1-5, doi: 10.23919/ICINS51816.2023.10168407.

[3] Savelyev, B & Solodov, S & Tropin, D. (2021). Formalizing and securing relationships on multi-task metric learning for IoT-based smart cities. Journal of Physics: Conference Series. 2094. 032062. 10.1088/1742-6596/2094/3/032062.

[4] Efimoff, Albert & Matveev, Philip. (2022). Искусственный интеллект для науки и наука для искусственного интеллекта. Voprosy filosofii / Akademiia nauk SSSR, Institut filosofii. 95. 10.21146/0042-8744-2022-3-93-105.

[5] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners. In Advances in Neural Information Processing Systems (NeurIPS), 2020.

[6] Aleksandr Smolin & Andrei Yamaev & Anastasia Ingacheva & Tatyana Shevtsova & Dmitriy Polevoy & Marina Chukalina & Dmitry Nikolaev & Vladimir Arlazarov, 2022."Reprojection-Based Numerical Measure of Robustness for CT Reconstruction Neural Network Algorithms," Mathematics, MDPI, vol. 10(22), pages 1-17, November. <https://ideas.repec.org/a/gam/jmathe/v10y2022i22p4210-d969599.html>

[7] Heewoo Jun, Alex Nichol (2023). Shap-E: Generating Conditional 3D Implicit Functions arXiv:2305.02463 [cs.CV]

[8] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. arXiv:2011.13456, 2020.

[9] Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. arXiv:arXiv:1907.05600, 2020b.

[10] Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. arXiv:2006.11239, 2020.

[11] Alex Nichol, Heewoo Jun, Prafulla Dhariwal, Pamela Mishkin, Mark Chen (2022). Point-E: A System for Generating 3D Point Clouds from Complex Prompts. arXiv:2212.08751 [cs.CV].

[12] Dhariwal, P. and Nichol, A. Diffusion models beat gans on image synthesis. arXiv:2105.05233, 2021.

[13] Ho, J. and Salimans, T. Classifier-free diffusion guidance. In NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications, 2021. URL https://openreview.net/forum?id=qw8AKxfYbI.

[14] Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations. arXiv:2011.13456, 2020

[15] Karras, T., Aittala, M., Aila, T., and Laine, S. Elucidating the design space of diffusion-based generative models. arXiv:2206.00364, 2022.

[16] Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv:2112.10741, 2021.

[17] Guibas. Learning represen tations and generative models for 3d point clouds. arXiv:1707.02392, 2017.

[18] Ben Poole, Ajay Jain, Jonathan T. Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. arXiv:2209.14988, 2022

[19] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations, 2022. URL https://arxiv.org/abs/2208.02801.

[20] Ajay Jain, Ben Mildenhall, Jonathan T. Barron, Pieter Abbeel, and Ben Poole. Zero-shot text-guided object generation with dream fields. arXiv:2112.01455, 2021.

[21] Nasir Mohammad Khalid, Tianhao Xie, Eugene Belilovsky, and Tiberiu Popa. Clip-mesh: Generating textured meshes from text using pretrained image-text models. arXiv:2203.13333, 2022

[22] Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. arXiv:2201.12204, 2022.

Исследование распознавания производственных дефектов на стальных поверхностях при помощи компьютерного зрения

А.Г. Ерещенко
кафедра инженерной кибернетики НИТУ «МИСИС»
Москва, Россия
n1804172@edu.misis.ru

Аннотация **– данная научная статья представляет собой исследование разработки системы контроля, основанной на совместном применении компьютерного зрения и нейронных сетей, с целью обнаружения дефектов на поверхности стальных изделий. Цель данного исследования – ответить на вопрос, достаточно ли, на сегодняшний день, высок уровень развития технологий компьютерного зрения и нейронных технологий в целом, для применения их в области классификации дефектов на стальных поверхностях. Чтобы дать ответ на поставленный вопрос в данной исследовательской работе рассмотрена свободно распространяемая сиамская сверточная нейронная сеть [1] с открытым исходным кодом, основанной на фреймворке PyTorch. В процессе исследования анализируется возможность применения данной модели для классификации дефектов на стальных поверхностях, как с изображений, заготовленных для обучения нейросети, так и тестирования обученной модели на данных, которые не участвовали в обучении с последующим сравнением полученных результатов.**

Ключевые слова **– Компьютерное зрение, Производство, Контроль производственных процессов, Контроль качества, PyTorch, Inception V3, OpenCV.**

I. ВВЕДЕНИЕ

В условиях все растущих скоростей на линиях и возрастающих требований к качеству продукции со стороны дистрибьюторов и потребителей традиционный подход к контролю технологических процессов при помощи «человеческого глаза» становится малоэффективным.

Ручной контроль качества все еще превалирует на многих видах производств, так как любое промышленное предприятие имеет целый арсенал обученных сотрудников и отточенных стандартов качества, на соответствие которым изделия проходят проверку. Но в условиях всё растущих уровней производства и потребления, а также требований, предъявляемых к качеству продукции возникает ощутимая потребность внедрять более быстрые и точные методы контроля за производством.

На сегодняшний день можно с уверенностью сказать, что автоматизированная визуальная дефектоскопия, основанная на компьютерном зрении, способна в значительной степени сократить непосредственное участие работников в процессе проверки качества на всех видах производственных линий, отведя человеку роль руководителя процесса.

Современные системы контроля качества способны провести подсчет объектов, снять измерения, проверить цвет, комплектность, наличие маркировок и штрихкодов, выявить дефекты и, при необходимости, сопоставить изделие с эталоном. Преимущества искусственного интеллекта [2] уже используются при навигации летательных аппаратов [3], системах распределения антропогенной нагрузки [4], распознавании текста [5]. Мощности систем компьютерного зрения помогают производствам решать задачи дефектоскопии исследуемых объектов точно, быстро и готовы повторять процедуру неограниченное количество раз.

Глубокие нейронные сети создают основу для обучения сложным характеристикам входных данных при большом количестве данных. В результате область компьютерного зрения также переходит от статистических методов к подходам на основе глубоких нейронных сетей.

Для обучения моделей компьютерного зрения хорошо зарекомендовали себя сверточные нейронные сети уже сейчас они широко используются в компьютерном зрении и демонстрируют отличные результаты Их способность эффективно извлекать признаки из изображений делает их отличным выбором для анализа и обработки визуальной информации.

Однако стоит заметить, подготовка данных и обучение моделей [6] нелёгкий процесс и требует больших объёмов вычислительных мощностей [7].

II. НАБОРЫ ДАННЫХ

Для обучения и тестирования рассматриваемой в данной работе нейронной сети использовались два набора данных взятые из отрытых источников. Несколько наборов данных призваны обеспечить объективность полученных после тестирования результатов. Оба набора данных представляют собой базы данных поверхностных дефектов, разделенные на 6 классов.

A. NEU-surface-defect-database [8,9]

Набор представляет собой базу данных поверхностных дефектов Северо-Восточного Университета (NEU) [8, 9]. Эта база данных состоит из шести классов поверхностных дефектов горячекатаной стальной полосы, а именно: вкатанная окалина (RS), пятна (Pa), трещины (Cr), ямочная поверхность (PS),

включения (In) и царапины (Sc).Всего в наборе данных 1800 полутоновых изображений, по 300 образцов для каждого из шести классов. Разрешение каждого образца изображения составляет 200x200 пикселей. Несколько образцов изображений из набора данных для каждого класса показаны на рис. 2. Изображения из набора данных имеют разную в освещенности, что создает дополнительные трудности для решения задачи распознавания изображений распознавания.

Эта вариативность приводит к большим различиям в образцах, принадлежащих к одному классу. Еще одна проблема, которую можно наблюдать из-за сходства изображений, принадлежащих к разным классам, как видно на рис. 1.

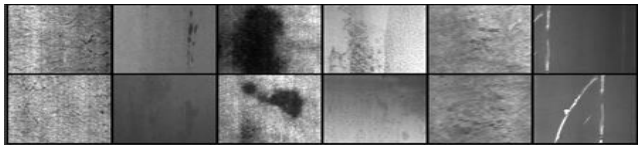


Рис. 1. Случайные выборки из набора данных NEU

Например, сходство изображений, относящихся к категориям "трещины" и “окалина” легко заметить.

B. Industrial Optical Inspection-database

Второй набор данных [10] представляет собой базу данных DAGM (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., немецкое отделение IAPR (Международной ассоциации распознавания образов)).

Набор данных используется непосредственно для проверки рассматриваемой нейросети, полученной в результате обучения на первом наборе данных. Данные манипуляции проверяют готовность нейросети к реальной работе. В набор входят 3450 изображений, которые по аналогии с первым набором, были разбиты на 6 классов.

Источник данных отмечает, что данные сгенерированы искусственно, но подходят для решения реальных проблем.

Несколько образцов изображений из второго набора данных показаны на рис. 2.

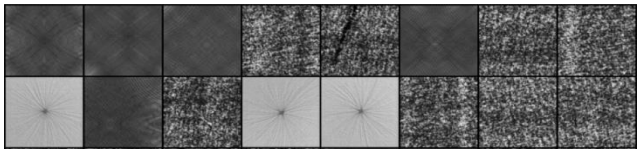


Рис. 2. Случайная выборка из набора данных DAGM

Чтобы преодолеть проблему ограниченного количества и разнообразия данных, существующие наборы данных были дополнены с помощью различных преобразований.

Каждое изображение в наборе данных случайным образом поворачивается вокруг своего центра. Угол поворота выбирается равномерно из множества углов $0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}$ (в радианах). Помимо этого, используются горизонтальные и вертикальные перевороты в данных, каждый с вероятностью 50%. Чтобы поддержать инвариантности модели к условиям освещенности, вводятся возмущения в яркости изображения. Случайная величина, выбранная из равномерного распределения в диапазоне [-10, 10], добавляется в изображение для этого возмущения. Уравнения (1), (2) определяют операцию возмущения.

$$\beta \sim U(-10, 10) \quad (1)$$

$$I_{out} = \max(\min(I_{in} + \beta, 255), 0) \quad (2)$$

В приведенных выше уравнениях I_{in} представляет собой входное изображение, I_{out} – выходное изображение. U (-10; 10) представляет собой равномерное распределение для выборки скалярного значения.

III. НЕЙРОСЕТЕВАЯ АРХИТЕКТУРА

Алгоритм обучения был реализован в образе системы one-shot-steel-surfaces [11].

Ключевая идея распознавания дефектов по одному изображению заключается в том, что при наличии изображения определенного класса, сеть должна быть способна распознать, принадлежат ли примеры-кандидаты к одному классу или нет. Сеть учится определять различия в характеристиках пары входных изображений в процессе обучения. На этапе вывода, обученная сеть может быть повторно использована только с одним примером изображения определенного класса, чтобы распознать, принадлежат ли данные-кандидаты к одному классу или нет.

Архитектура сиамской сети [1], используемая в данной работе, показана на рис. 3.

Эта модель обучена для получения хорошего представления дефектов на стальных поверхностях. Модель после обучения должна быть способна распознавать несколько дефектов на при наличии одного примера каждого дефекта.

На рис. 3 два модуля сети идентичны и имеют одинаковые веса. Каждый модуль можно рассматривать как параметрическую функцию весов 0, заданную $f0 : R^N \rightarrow R^n$ и N >> n. Входной сигнал высокой размерности (изображение) R^N сводится к выходу, который представляет собой закодированный вектор меньшей размерности n. В данном случае N = 100x100 и n = 5. Так же следует обратить внимание на то, что выходы двух модулей из слоев с размером n = 5 называются закодированными векторами $f_0(x1)$ и $f_0(x2)$.

Конечным результатом архитектуры является евклидово расстояние между этими закодированными векторами.

Входными данными для модели являются одноканальные канальное или полутоновое изображение пар x1 и x2. Каждый модуль идентичный, имеет три конволюционных слоя с количеством карт признаков 4, 8 и 8 слева направо соответственно, размером 100x100 каждая. За конволюционными слоями следуют три полностью связных слоя размером 500, 500 и 5 соответственно. Размер ядра 3x3 используется для сверток с шагом 1. Функция активации ReLU используется для выходных карт признаков каждого слоя.

Для обучения использовалась функция контрастных потерь [10, 11]. Уравнение (3) описывает функцию потерь L (). Функция потерь параметризуется весами нейронной сети 0 и обучающей выборкой i. Обучающий образец из набора данных представляет собой кортеж $(x1; x2; y)^i$, где x1 и x2 - пара изображений, а метка y

равна 1, если x1 и x2 принадлежат к одному классу, и 0 в противном случае.

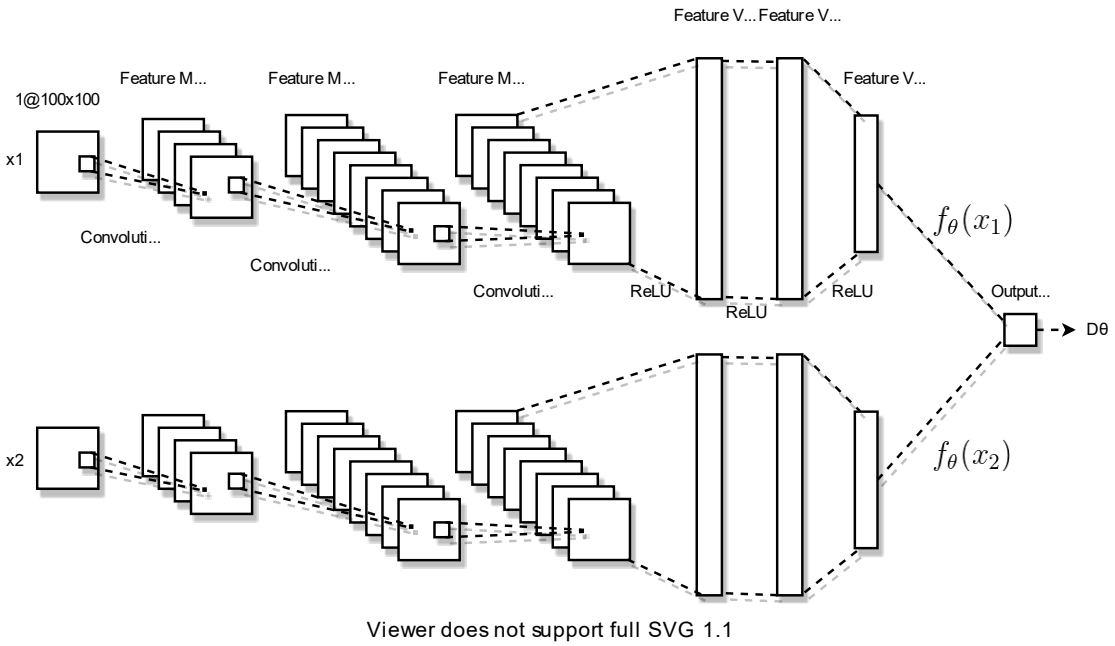


Рис. 3. Нейросетевая архитектура

$$L(0,(x_1,x_2,y)^i = y\frac{1}{2}D_{0,i}^2 + (1-y)\frac{1}{2}(\max\{0,m-D_{0,i}^2\})^2 \quad (3)$$

Первый член правой части (RHS) уравнения (3) накладывает затраты на сеть, если пара входных изображений x1 и x2 принадлежит одному классу, т.е. y = 1. Второй член отбрасывает выборку если входная выборка принадлежит к разным классам т.е y = 0. m > 0 - это маржа, значение которой постоянно. Значение $D_{0,I}$ объясняется в уравнении (4).

$$D_{0,i} = ||f_0(x_1) - f_0(x_2)||_{2,i} \quad (4)$$

Уравнение (4) представляет собой евклидово расстояние между n-мерными выходами модулей нейронной сети для входного изображения x1 и x2 в выборке i-го набора данных. Для i-й выборки с y = 1 второй член в уравнении (4) равен нулю. Таким образом, величина потерь в этом случае прямо пропорциональна квадрату расстояния между f_0(x1) и f_0(x2). Целью является минимизация потерь, весовые коэффициенты сети обучаются таким образом, чтобы уменьшить расстояние между закодированными векторами входных образцов x1 и x2. Интуитивно это можно понять так, что модель учится определять похожи ли два входных изображения. С другой стороны, если входной образец имеет метку y = 0, то первый член в RHS обнуляется. Если y = 0 и $D_{0,I}$ > m, модель не штрафуется. Штраф применяется только в том случае, если евклидово расстояние между f(x1) и f(x2) меньше, чем заданная маржа m. Задача в этом случае состоит в том, чтобы отодвинуть закодированные векторы f_0(x1) и f_0(x2) друг от друга в n мерном пространстве и сделать расстояние между ними больше, чем m. Можно рассматривать второй член в функции потерь как модель, обучающуюся понимать различия между x1 и x2 которые принадлежат к разным классам. В результате использования этой функции потерь сиамская сеть не только учится оценивать сходство входной пары изображений, но и оценивать значения потерь для несходных пар с ненулевым вторым членом и позволяют избежать свертывания модели к постоянной функции.

Для подробного математического объяснения контрастных потерь авторы просят читателей обратиться к статье [8].

IV. ОБУЧЕНИЕ И ТЕСТИРОВАНИЕ

Для тестирования работоспособности выбранной нейронной модели сперва её требуется обучить. В данном случае переменная, настройка которой напрямую повлияет на успешность обучения это выбор количества эпох для тренировки нейросети. Эпоха в машинном обучении представляет собой один проход через все обучающие примеры сети. Во время каждой эпохи модель получает все данные для обучения, обрабатывает их и корректирует веса в соответствии с обнаруженными ошибками. Обычно обучение нейронной сети включает в себя множество эпох, чтобы гарантировать, что модель достаточно хорошо обучена на всех данных. Соответственно, чем больше эпох обучения нейронной сети, тем больше модель имеет возможность "почувствовать" структуру и закономерности в данных, что может привести к более точной и эффективной работе модели. Однако, есть вероятность переобучения модели, если количество эпох будет слишком большим, в результате чего модель начнет "запоминать" обучающие данные, но не сможет делать точные предсказания на новых данных. Поэтому оптимальное количество эпох

обучения зависит от конкретной ситуации и требует баланса между обучением модели и предотвращением переобучения. Для обучения было решено установить 100 эпох, как оптимальное число для получения достоверных результатов, и не больших временных затрат

Нейросетевая модель была обучена с использованием набора данных NEU [9] по поверхностным дефектам. Значения гиперпараметров, использованные для обучения приведены в таблице 1.

ТАБЛИЦА 1. Значения параметров при обучении сети

Параметр	Значение
Batch size	32
Количество эпох	100
Шаг обучения	5e-4
Margin m	2
Функция оптимизатор	Adam [12]
Параметры функции	(0,9, 0.999)

Набор данных NEU [9] был разделен на два набора для одномоментного распознавания. Обучающий набор состоял из трех классов, а именно: катанные окалины, пятна, включения. Остальные классы - трещины, ямки и царапины были показаны сети на этапе тестирования для однократного распознавания. Выборки данных при обучении выбирались случайным образом. При выборке пары изображений выбирались два изображения из одной категории с вероятностью 0.5 с соответствующей меткой y = 1. Аналогично, изображения были выбраны из двух различных категорий с остаточной вероятностью 0.5 с меткой y=0. Этот кортеж из пары изображений и метки (x1; x2; y) затем дополняется преобразованиями, описанными в разделе 4. Перед передачей изображения в сеть значения пикселей каждого изображения были нормализованы и попали в диапазон [-1; 1].

Кривые обучения и проверки для оптимизации модели, обученной на наборе данных из 900 образцов изображений, дополненных преобразованиями, как описано в разделе 4, показаны на рис. 3.

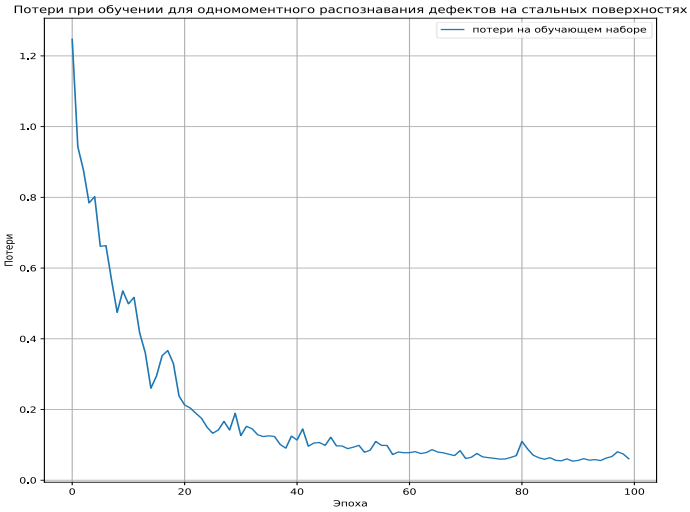


Рис. 3. – Кривая потерь при обучении модели

Эксперименты проводились на платформе Intel-i5 с 16 ГБ оперативной памяти и NVIDIA RTX 3050. Обучение с набором данных по поверхностным дефектам прошло довольно быстро. На обучение этой архитектуры с нуля ушло около часа.

Обучение проводилось в течение 100 эпох с параметром Batch = 32. Здесь мы видим тенденцию к уменьшению потерь по мере увеличения количества эпох. Это связано с тем что в процессе обучения модель осознает характерные черты эталонного изображения и изображения-кандидата таким образом, потери, накопленные в процессе обучения, уменьшаются. На этапе тестирования изображения выбирались случайным образом. Эти изображения принадлежат к другому набору классов, которые не были показаны сети во время обучения. На рисунке 4 показаны некоторые результаты работы сиамской сети [1], оцененные в ходе тестирования.

Результаты представлены с названиями классов и изображений, а также с оценкой несхожести пары изображений. Показатель несходства - это значение уравнения (2) для истинного изображения (x1) и изображения-кандидата (x2). Из этого рисунка видно, что изображения, принадлежащие к разным категориям, имеют большее значение балла несходства по сравнению с изображениями, принадлежащими к одной и той же категории. У изображений из несхожих классов балл больше, чем значение маржи m, используемой в функции контрастных потерь. Из этого наблюдения можно сделать вывод, что архитектура нейронной сети способна эффективно понимать сходства и различия между признаками входных образцов.

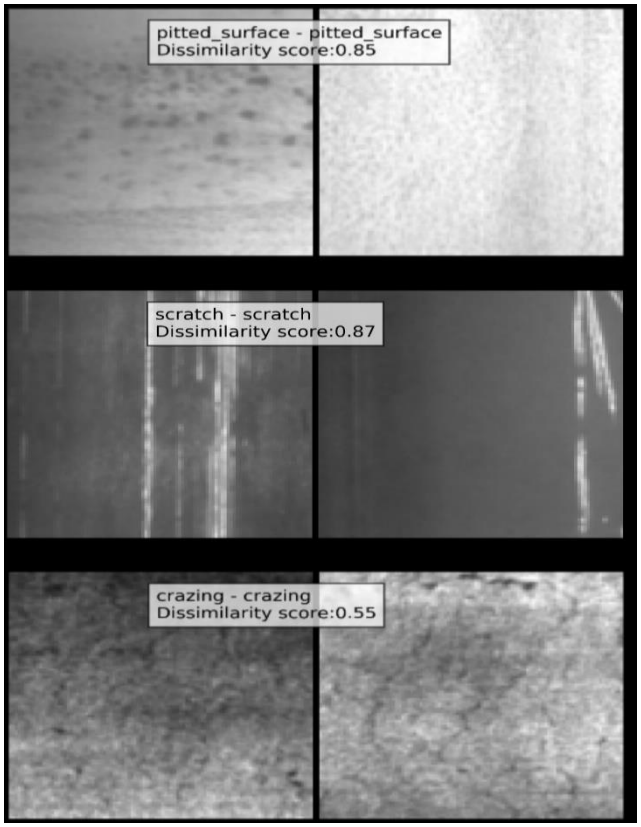


Рис. 4. – Результаты полученные на этапе обучения

Стоит упомянуть, что данная модель тестировалась на различных алгоритмах обучения.

Сравнивались результаты одномоментного распознавания с помощью алгоритма классификации K-nearest neighbor (KNN) и архитектуры фидфорвардной конволюционной нейронной сети.

Алгоритм был использован для классификации изображений дефектных поверхностей. KNN было показано по одному экземпляр изображений из каждого класса набора данных, и его Точность теста оценивалась по доле правильно классифицированных тестовых экземпляров. В качестве входных данных использовались необработанные изображения, а евклидово расстояние между изображениями использовалось в качестве метрики в этом алгоритме для классификации изображений-кандидатов в определенной категории.

Помимо этого, подход, исследуемый в данной работе, сравнивался с классификатором CNN. CNN, который использовался для этого сравнения, имеет аналогичную архитектуру, как и один из модулей сиамской сети [1]. На вход сети подается одноканальное изображение. Выходными данными сети являются вероятности принадлежности входного изображения к одному из классов принадлежности входного изображения к одному из шести классов из набора данных о поверхностных дефектах. Сеть состояла из трех конволюционных слоев с картами признаков 4, 8 и 8 соответственно. Размер каждой карты признаков составлял 100x100. Размер ядра 3 использовался для свертки с шагом 1 в этих слоях. За третьим конволюционным за третьим сверточным слоем следовали два полностью связанных слоя размером 500 каждый. Выходной слой состоял из 6 нейронов. В качестве функции активации использовалась функция активации ReLU, за исключением выходного слоя, который сигмоидальной функции активации для представления вероятностей классов входного изображения. Обучающий набор состоял из 80 % набора данных, а оставшиеся данные использовались для проверки и тестирования этой сети. Категориальная потеря перекрестной энтропии была для обучения CNN вместе с оптимизатором Адама. Эта Сеть обучалась в течение 120 эпох с размером партии 128.

В таблице 2 приведены результаты тестирования каждого метода на наборе данных о дефектах стальной поверхности.

ТАБЛИЦА 2. Результаты тестирования различных алгоритмов обучения [9]

Алгоритм	Точность (%)
K-nearest neighbor(KNN)	28.22
Сиамская нейросеть	83.22
CNN	93.24

Из таблицы 2 видно, что алгоритм KNN работает плохо и демонстрирует низкую производительность на этапе вывода фазе. Очевидно, что его невозможно использовать в реальных сценариях, поскольку он не оптимизирован для хорошего представления признаков данных, а также метрика евклидова расстояния не является подходящей функцией для количественной оценки соответствия между данных изображений высокой размерности [13]. Несмотря на то, что CNN имеет более высокую производительность, следует также отметить, что для одномоментного распознавания предъявлялся только один образец изображения из новой для получения наблюдаемой производительности, в отличие от 80 % данных из каждой категории, использованных для обучения CNN.

Набор обучающих данных для обеих моделей состоял 80% набора данных NEU из всех его шести классов и оставшиеся 6 классов данных были использованы для проверки и тестирования. В этой таблице представлены точность тестирования обеих моделей. Все гиперпараметры обучения и функции потерь оставались такими же, как описано ранее в этом разделе для соответствующих нейросетевых моделей. Из результатов, приведенных в таблице 3, видно, что CNN и сиамская имели конкурентоспособную производительность при обучении на идентичных данных из набора данных по поверхностным дефектам NEU. Результаты в этой таблице также свидетельствуют о том, что сиамская сеть сходится к производительности CNN по мере увеличения размера набора данных. используемых для ее обучения.

Основываясь на результатах, полученных с помощью сиамской сети для одномоментного распознавания, можно сказать, что этот подход имеет потенциал для простого и быстрого развертывания на реальных производственных площадках в случае

ограниченного количества обучающих данных. Учитывая постоянно растущие производственные требования и растущие требования к автоматизации контроля качества, это может стать подходящим приложением для ситуаций, когда аннотирование данных затруднено или доступность данных ограничена.

Для подтверждения точности классификации дефектов был произведён второй запуск. Во время второго запуска использовался второй набор данных [10], который не участвовал в обучении нейросети, но при этом данный набор так же, как и первый был разделен на 6 классов дефектов, описанных ранее.

Данный эксперимент нужен для того, чтобы проверить обученную нейронную сеть на корректность классификации дефектов с изображений, которых нейросеть не «видела» в процессе обучения. То есть, для успешности теста, нейросеть должна так же распознать и классифицировать дефекты с изображений, как и при обучении.

На рисунке 5 можно наблюдать кривую потерь при тестировании модели на наборе данных 2.

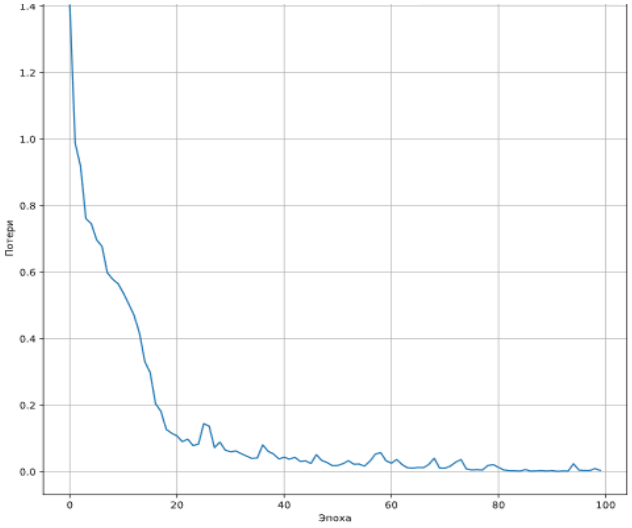


Рис. 5. График потерь при тестировании модели

Исходя из данного графика, можно заметить, что модель справилась с задачей несколько лучше и стабильнее, чем с данными, используемыми при обучении. Вероятнее всего, это объясняется тем, что набор данных 2 [10] является искусственно сгенерированным, что несколько облегчило задачу при распознавании дефектов. Тем не менее, исходя из результатов эксперимента можно сделать вывод о том, что модель успешно справляется с задачей классификации дефектов на стальных поверхностях. Но, стоит отметить, что данная модель требует особых манипуляций при подготовке данных, таких как разделение наборов данных на определенные классы дефектов.

V. ЗАКЛЮЧЕНИЕ

В данной работе было выполнено тестирование свободно распространяемого классификатора дефектов стальных поверхностей, предложенный авторами работы [11], который представляет собой модель, обученную методами искусственного интеллекта [14] на основе компьютерного зрения и сиамской сверточной нейронной сети [1].

Предварительно, перед тестированием данная модель была обучена на первом наборе данных [9], который был заготовлен авторами [11]. Затем данное решение было протестировано на новом, самостоятельно подобранном из открытых источников наборе данных [10]. Данный эксперимент выполнялся для того, чтобы проверить обученную нейронную сеть на корректность классификации дефектов с изображений, которых нейросеть не «видела» в процессе обучения и оценить её готовность к работе в «полевых» условиях. То есть, для успешности тестирования, нейросеть должна так же хорошо распознать и классифицировать дефекты с изображений из набора данных 2, как и при обучении на наборе данных 1.

После проведения тестов на разных наборах данных и сравнения результатов работы сети был сделан вывод о том, что данная модель способна эффективно классифицировать как исходные данные, заложенные при обучении нейронной сети, так и новые данные, которые до этого не участвовали при её обучении, что является показателем её эффективности и значимым аргументом к внедрению систем классификации дефектов на её базе в реальные производства.

ЛИТЕРАТУРА

[1] Siamese Neural Networks for One-Shot Image Recognition Gregory R. Koch, 2015, URL: https://www.semanticscholar.org/paper/Siamese-Neural-Networks-for-One-Shot-ImageKoch/f216444d4f2959b4520c61d20003fa30a199670a

[2] Anokhin, K.V., Novoselov, K.S., Smirnov, S.K., Efimov, A.R., & Matveev, P.M. (2022). AI for Science and Science for AI. Voprosy Filosofii.

[3] Ali, B., Sadekov, R.N. & Tsodokova, V.V. A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy Navig. 13, 241–252 (2022). https://doi.org/10.1134/S2075108722040022

[4] Y. S. Chernyshova, B. I. Savelyev, S. V. Solodov, S. V. Pronichkin,“Applying distributed ledger technologies in megacities to faceanthropogenic burden challenges,” in IOP Conference Series: Earthand Environmental Science, 2022, vol. 1069, no. 1. doi:10.1088/1755-1315/1069/1/012028.

[5] D. V. Polevoy, P. A. Kulagin, A. S. Ingacheva, Zh. V. Soldatova, M.V. Chukalina, D. P. Nikolaev, V. V. Arlazarov, “From tomographicreconstruction to automatic text recognition: the next frontier task forthe artificial intelligence,” Fifteenth International Conference onMachine Vision (ICMV 2022), 2023, vol. 12701.doi:10.1117/12.2680132.

[6] Николенко С., Кадурин А., Архангельская Е. Глубокое обучение.Погружение в мир нейронных сетей. – СПб. : Питер, 2018. – 480 с. :ил. – ISBN 978-5-496-02536-2.

[7] В.В. Селякин. Компьютерное зрение. Анализ и обработка изображений. Лань. Специальная литература. 2019. 978-5-81-143368-1-152

[8] He, Y., Song, K., Dong, H., Yan, Y., 2019a. Semi-supervised defect classification of steel surface based on multi-training and generative adversarial network. Optics and Lasers in Engineering 122, 294–302.

[9] NEU-surface-defect-database https://www.kaggle.com/datasets/rdsunday/neu-urface-defect-database/

[10] DAGM (Deutsche Arbeitsgemeinschaft für Mustererkennung e.V., немецкое отделение IAPR (Международной ассоциации распознавания образов)) - https://hci.iwr.uni-heidelberg.de/content/weakly-supervised-learning-industrial-optical-inspection

[11] One-Shot Recognition of Manufacturing Defects in Steel Surfaces - https://www.researchgate.net/publication/342400903_One-Shot_Recognition_of_Manufacturing_Defects_in_Steel_Surfaces

[12] Kingma, D.P., Ba, J., 2014. Adam: метод стохастической оптимизации. arXiv preprint arXiv:1412.6980 .

[13] Хадселл, Р., Чопра, С., ЛеКун, Й., 2006. Снижение размерности путем обучения инвариантному отображению, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), IEEE. pp. 1735-1742.

[14] Рассел С., Норвиг П. Искусственный интеллект: современныйподход, 2-е изд.. : Пер. с англ. - М. : Издательский дом “Вильямс”,2007. - 1408 с.

Исследование возможности детектирования дорожных знаков

В. О. Кирвяков
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m1809131@edu.misis.ru

***Аннотация*— В данном исследовании подвергнуты анализу несколько решений, основанных на коммерческом программном обеспечении, и проведено сравнение возможностей распознавания дорожных знаков, в разные погодные условия, время суток, а также перекрытие и искажение дорожных знаков. Работа выполнена на данных, полученных с камеры, передвижной лаборатории, предоставленного компанией, а также дополненных собственными снимками. В работе использовались одни из самых распространённых в области детекции и классификации изображений нейронные сети – Yolo и VGG.**

Ключевые слова — компьютерное зрение, детекция, дорожные знаки, распознавание дорожных знаков, беспилотные автомобили, YOLO, VGG

I. Введение

В современном обществе, где автономные и беспилотные системы становятся все более важными компонентами транспортной инфраструктуры, вопросы обеспечения безопасности и эффективности движения автономных транспортных средств выходят на передний план [1]. Одним из ключевых аспектов успешной реализации таких систем является надежное и точное детектирование дорожных знаков. Дорожные знаки представляют собой важные элементы дорожной инфраструктуры, предназначенные для регулирования движения транспорта и обеспечения безопасности участников дорожного движения.

В связи с тем, что современные технологии в области глубокого обучения [2], особенно нейронные сети, продемонстрировали впечатляющие результаты в обработке и анализе изображений [3, 4], представляется весьма перспективным применение этих методов для задачи детектирования дорожных знаков. Нейронные сети, обученные на соответствующих наборах данных, обладают потенциалом высокой точности распознавания и способны адаптироваться к разнообразным условиям дорожной обстановки.

В контексте развития технологий глубокого обучения [5] вопрос выбора оптимальных архитектур нейронных сетей для задачи детектирования дорожных знаков становится весьма актуальным [6]. Две из наиболее распространенных архитектур – YOLO и VGG – представляют собой различные подходы к решению задач компьютерного зрения [7, 8].

Архитектура YOLO отличается от других подходов тем, что позволяет проводить детекцию объектов на изображении однократным просмотром всего изображения. Сложность данной сети заключается в способности точно выделить множество различных объектов на разреженных изображениях дорожных сцен.

В свою очередь, архитектура VGG [9] привлекает внимание своим глубоким строением и серией сверточных слоев. Этот подход обеспечивает высокую точность классификации, но может потребовать больше вычислительных ресурсов в сравнении с YOLO [10].

Настоящее исследование направлено на сравнение эффективности нейронных сетей YOLO и VGG в контексте детектирования дорожных знаков. Анализ и сравнение этих двух архитектур позволят выявить их преимущества и ограничения в рамках данной задачи, а также предоставят важные инсайты для оптимизации систем детектирования в автономных транспортных средствах.

II. Наборы данных

С целью проведения процессов обучения и тестирования рассматриваемых в данном исследовании нейронных сетей были вовлечены различные наборы данных, включая как локально собранные авторами, так и ограниченные доступом, предоставленные компанией, специализирующейся на распознавании дорожных объектов. Рассмотрим данные использованные наборы более детально. Для создания набора данных были использованы актуальные методы сбора информации [11].

A. Коммерческий (закрытый) набор данных

Обширный архив данных, предоставленный организацией, основывается на видеозаписях дорожных сценариев, зафиксированных на дорогах России в различных регионах. Этот набор данных включает в себя аннотированные последовательности видео, зафиксированные под различным освещением и при различных погодных условиях, с использованием передвижной лаборатории [12] (см. рисунок 1). Кроме того, в наборе данных представлены все виды дорожных знаков РФ – всего 7 классов (рисунок 3): *предупреждающие знаки, знаки приоритета, запрещающие знаки, предписывающие знаки, знаки особых предписаний, информационные знаки, знаки сервиса.*

Рис. 1. Передвижная лаборатория НПО Регион.

На рисунке 2 отображены сложные для распознавания знаков ситуации:

- Изменение тонов цвета, обусловленные воздействием атмосферных факторов
- полное или частичное перекрытие другими объектами;
- неполная форма дорожного знака;
- Отражение фар передвижной лаборатории и других автомобилей;
- Перспективные искажения
- Размытие знаков на высокой скорости

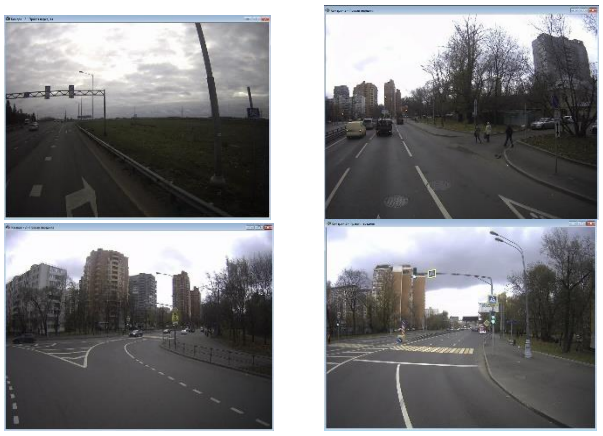


Рис. 2. Примеры сложных ситуаций

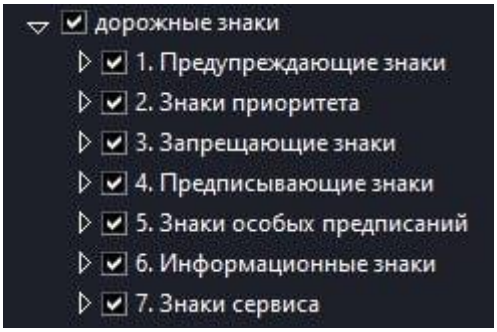


Рис. 3. Примеры классов дорожных знаков: а) Предупреждающие знаки, б) Знаки приоритета, в) Запрещающие знаки, г) Предписывающие знаки, д) Знаки особых препятствий, е) Информационные знаки, ж) Знаки сервиса

B. Собственный набор данных

В данной статье так же используется локальный набор данных, собранный автором работы, который состоит из фотографий знаков в ночное время суток, так как в датасете НПО Регион не содержится примеров дорожных знаков в ночное время, ввиду не надобности данных снимков для компании.

Данные получены с видеорегистраторов и фотокамеры смартфона (рисунок 4).

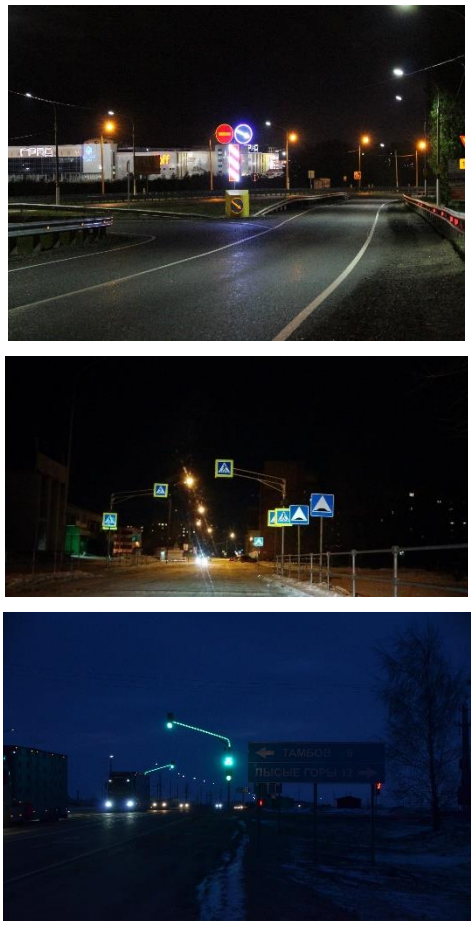


Рис. 4. Примеры кадров в ночное время суток.

III. Нейросетевые архитектуры

A. YOLOv3

В данном исследовании [13], рассматривается задача распознавания дорожных знаков. Предложено использование детектирующей нейронной сети, а также классификация знаков по категориям. Основным объектом интереса в текущем исследовании является нейросетевой аспект, связанный с детектированием и распознаванием кадров в кадре. В работе применяется YOLOv3, архитектура которой представлена на рисунке 6. Модель обучена для обнаружения и распознавания дорожных знаков. Выходной слой нейронной сети содержит сорок два класса, объединяющих различные виды дорожных знаков. Авторы отмечают, что это не исчерпывающий список дорожных знаков, встречающихся на дорогах Российской Федерации. Архитектура, описанная авторами, была адаптирована для учета всех необходимых классов.

Для обучения нейронной сети использован дополненный ночными кадрами датасет от НПО "Регион". Обучение проведено на примерно 5000 кадрах с размеченными дорожными знаками, а тестирование выполнено на отдельном датасете из примерно 22000 кадров с 6950 знаками. Функция потерь mAP [14] (средняя площадь под кривой точность-полнота) использована в качестве критерия оценки.

YOLOv3 обучалась на 1500 батчах, в каждом из которых содержалось 64 изображения, с постоянным learning rate. Размер изображений составлял 608x608 пикселей, что является компромиссом между скоростью и качеством. В процессе обучения также использовались методы аугментации, такие как изменение оттенка, насыщенности, экспозиции, а также батч-нормализация. Кроме того, каждые 10 батчей производилось изменение разрешения изображений с 608x608 на разрешения, кратные 32, для повышения устойчивости модели к различным масштабам. Так же проведена работа по балансированию обучающей выборки [15].

Рис. 5. Пример работы YOLOv3 в задаче детектирования и распознавания дорожных знаков

B. VGG

Архитектура VGG [16] часто используется для задач классификации изображений и представлена на рисунке 7, но ее можно адаптировать и для задачи детектирования объектов, таких как дорожные знаки.

Основные черты архитектуры VGG [17] для детектирования дорожных знаков могут включать следующие этапы:

1. Входной слой: Изображение дорожного участка или кадра подается на вход нейронной сети.
2. Сверточные слои: используются несколько последовательных сверточных слоев с небольшими ядрами (обычно 3x3) для извлечения различных признаков из входного изображения. Эти слои помогают выделять узоры и характеристики изображения.
3. Подвыборка: после каждого сверточного слоя может использоваться слой подвыборки, такой как слой субдискретизации (max-pooling),

который уменьшает размер признаковых карт, сохраняя наиболее важные признаки.

4. Полносвязные слои: выходы последнего сверточного слоя передаются через один или несколько полносвязных слоев, которые обычно используются для классификации. В контексте детектирования объектов эти слои могут быть адаптированы для выдачи более детальных предсказаний о местоположении объектов.
5. Выходной слой: выходной слой генерирует предсказания, включая вероятности присутствия различных классов дорожных знаков и информацию о их местоположении (ограничивающие прямоугольники).
6. Функция потерь: для обучения сети используется функция потерь, которая оценивает разницу между предсказанными значениями и истинными метками. В задаче детектирования это может включать в себя компоненты, связанные с классификацией и локализацией объектов.
7. Обучение и дообучение: сеть обучается на размеченном наборе данных дорожных знаков. При необходимости сеть может быть дообучена на специфическом наборе данных для улучшения ее способности обнаружения конкретных дорожных знаков. Архитектура VGG обеспечивает глубокое извлечение признаков, что может быть полезным для выделения характерных черт дорожных знаков при их детектировании.

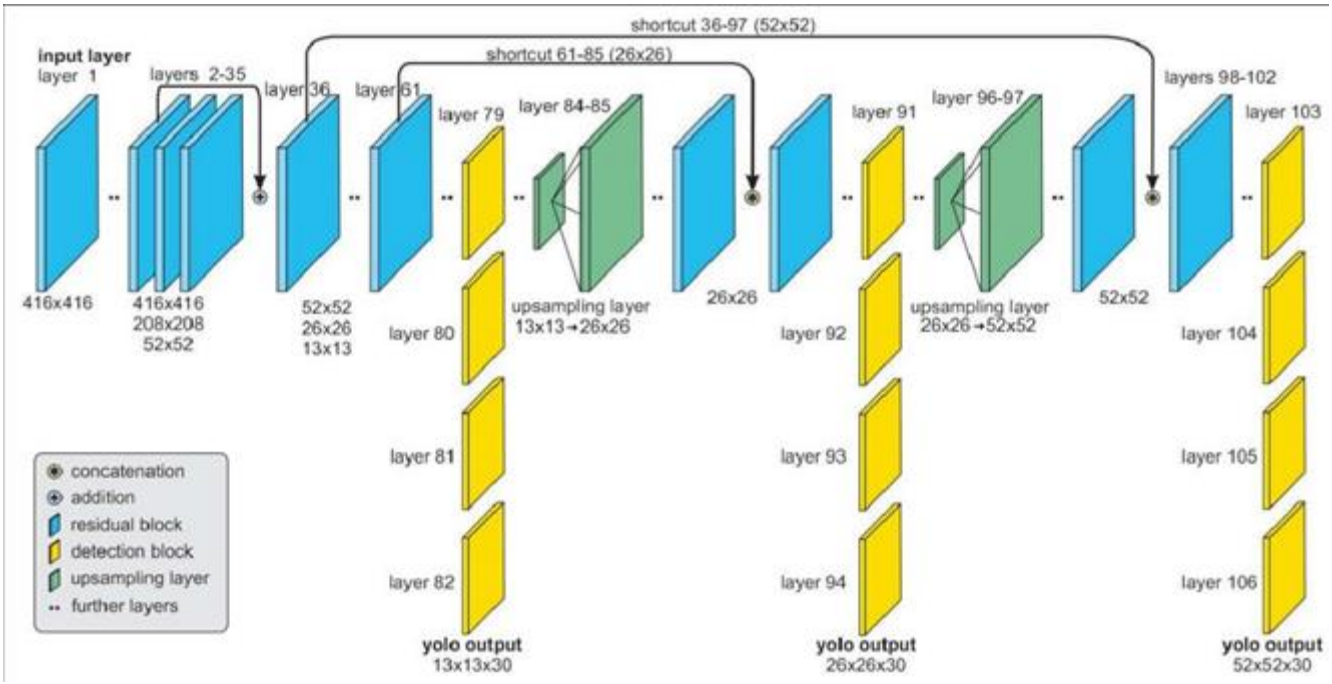


Рис. 6. Архитектура YOLOv3

VGG обучался на том же датасете что и вышеприведенный YOLOv3.

IV. Сравнение

Сравним два описанных подхода. Для сравнения используется отдельно собранный набор данных – 22063 изображения с 6950 размеченными дорожными знаками. Качество работы двух подходов складывается из качества работы, локализующей и классифицирующей частей. Оценка локализации производится при помощи расчёта меры Жаккара (Intersection over Union, IoU) для каждой детекции.

Введём следующие величины:

Разработка конечной системы велась в две стадии:

- TP – детектор верно локализовал дорожный знак и определил его класс.
- FP – детектор нашёл дорожный знак там, где его нет, или не верно определил его класс.
- FN – детектор не нашёл дорожный знак, хотя он есть и для него есть разметка.

Стоит отметить, что TN в данном случае не определена, так как это величина означает то, что детектор не определил дорожный знак, где его действительно нет. По введённым величинам строятся такие функции оценок, как:

- $Precision = \frac{TP}{TP+FP}$ – сколько раз детектор нашёл дорожный знак, где он действительно есть, по отношению к общему числу предсказанных знаков;
- $Recall = \frac{TP}{TP+FN}$ – сколько дорожных знаков нашёл детектор из действительно присутствующих в кадрах;
- $F1 = 2\frac{Precision \cdot Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN}$ – оценка баланса между точностью (precision) и полнотой (recall).

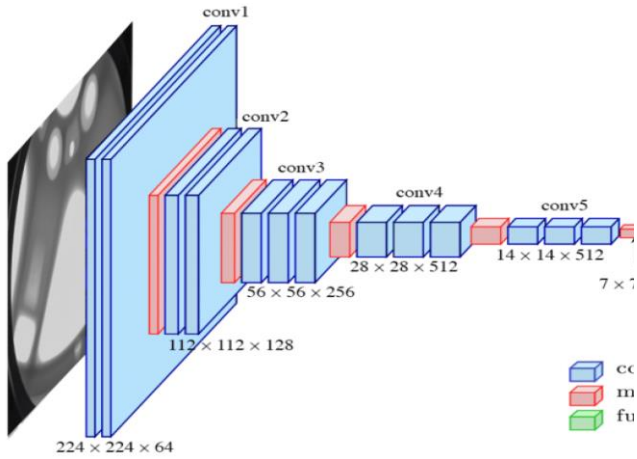


Рис. 7. Архитектура VGG

Таблица 1 отображает количественные оценки для двух подходов.

ТАБЛИЦА I. Оценка детектирующей части

	YOLOv3	**VGG**
TP	6537	5747
FP	244	1034
FN	169	322
Precision	0.96	0.84
Recall	0.97	0.45
F1	0.96	0.59

Как видно из таблицы детектор YOLOv3 имеет значительно более высокие показатели, что означает, что он намного больше находит действительных дорожных знаков и намного меньше ошибается, детектируя не относящееся к делу окружение.

В оценку классифицирующей части включены все объекты, которые входят в множество TP локализующей части. Классифицирующая нейронная сеть VGG второго подхода на выходе имеет 7 классов, так же, как и классификатор YOLOv3. Связи с этим матрицы ошибок имеют одинаковые размеры. На рисунке 10 отображена матрица ошибок и отчёт о классификации, содержащий precision, recall и F1-меру [18, 19], для нейросети VGG. Здесь номера классов 1–7 означают классы дорожных знаков: *предупреждающие знаки, знаки приоритета, запрещающие знаки, предписывающие знаки, знаки особых предписаний, информационные знаки, знаки сервиса*, соответственно.

VGG

	1	2	3	4	5	6	7
1	821	57	17	89	1	11	0
2	24	801	0	30	28	6	9
3	23	6	466	51	0	2	55
4	51	28	37	1360	77	0	29
5	0	54	0	17	655	21	17
6	9	5	48	71	0	575	0
7	18	9	47	38	49	0	1069

Рис. 8. Матрица ошибок и численная оценка работы VGG

Как можно видеть, все классы распознаются с не очень высоким значением F1-меры, но при этом распознались все классы. Равномерное распределение метрик качества можно объяснить хорошо составленной выборкой. Что касается точности детектирования и распознавания, по этим результатам видно, что данная архитектура нейронной сети плохо подходит к нашей задаче.

Классифицирующая часть YOLOv3 на выходе имеет так же 7 классов. Поэтому можно определить некоторую переходную матрицу ошибок, которая содержала бы 7 реальных классов. Рассмотрим эту матрицу подробнее (рисунок 8).

Численные оценки классификации этой нейросети имеют значения, близкие к 100% (рисунок 9). Такую сильную с предыдущей нейронной сетью разницу в качестве можно объяснить несколькими обстоятельствами: устойчивость YOLOv3 к таким искажениям как: перепады света, перспективные искажения, большее число тестовых картинок, так как более качественный детектор смог правильно локализовать большее число дорожных знаков, которые затем и классифицировались.

Сравнивая классификаторы YOLOv3 и VGG, можно заметить, что все классы, представленные в собранном датасете распределены равномерно. YOLOv3 распознаёт эти классы лучше из-за упомянутых обстоятельств. Сбалансированность распознанных классов ещё раз показывает важность составления репрезентативной, широкой и сбалансированной обучающей выборки.

YOLOv3

	1	2	3	4	5	6	7
1	936	0	20	37	0	3	0
2	14	865	0	18	0	1	0
3	8	0	544	49	0	2	0
4	15	0	4	1560	0	0	3
5	0	0	0	0	764	0	0
6	11	0	0	0	0	697	0
7	2	0	57	0	0	0	1171

Рис. 9. Матрица соответствий YOLOv3.

V. Заключение

Были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые

нейронные сети. Приведены два подхода к детектированию – локализации и классификации – дорожных знаков: модернизация YOLOv3, предложенная и обученная авторами работы [13], в которой решалась задача определения класса найденных дорожных знаков, и VGG [17], обученная на том же наборе данных. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведённые подходы были сравнены на собранном автором датасете – датасет НПО Регион и набор фотографий знаков в ночное время суток. Отдельно были оценены качество локализации и классификации знаков. По полученным данным очевидно, что нейронная сеть YOLOv3, подготовленная авторами работы [13], имеет сильное преимущество перед альтернативным подходом, что было обнаружить достаточно странно, так как VGG в среднем показывает результаты лучше, чем YOLOv3. Притом сравнивались именно конкретные модели, обученные на схожем датасете. Для сравнения архитектур в целом нужны фиксированные наборы и процессы обучения и тестирования.

Литература

[1] “Driverless cars (global market)”, available at: https://www.tadviser.ru/index.php/Статья:Беспилотные_автомобили_(мировой_рынок) (Accessed: December 25, 2022).

[2] Efimoff, Albert & Matveev, Philip. (2022). Искусственный интеллект для науки и наука для искусственного интеллекта. Voprosy filosofii / Akademiia nauk SSSR, Institut filosofii. 95. 10.21146/0042-8744-2022-3-93-105.

[3] Mask R-CNN: архитектура современной нейронной сети для сегментации объектов на изображениях. – Режим доступа: https://habr.com/ru/post/421299. – (Дата обращения 17.07.2019).

[4] Прэтт У. Цифровая обработка изображений. Т.1 и 2 М.: Мир. 1982 790 с.

[5] Yakovlev, A. & Kondybayeva, A. & Solodov, S.. (2019). Intelligent System for Collecting, Analyzing and Managing Data in the Field of Medicine. 1-6. 10.1109/WECONF.2019.8840588.

[6] Zhu Y., Zhang C., Zhou D., Wang X., Bai X., Liu W. Traffic sign detection and recognition using fully convolutional network guided proposals. Neurocomputing, 2016, vol. 214, pp. 758–766. doi: 10.1016/j.neucom.2016.07.009.

[7] Deep Residual Learning for Image Recognition / Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun // CoRR. — 2015 — Vol. abs/1512.03385. — 1512.03385.

[8] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 779–788, 2016.

[9] Pawe Staszewski, Maciej Jaworski, Jinde Cao, Fellow, IEEE and Leszek Rutkowski, Fellow “A new approach to descriptors generation for image retrieval by analyzing activations of deep neural network layers”, IEEE.

[10] Hoang, Lee, "An Evaluation of VGG16 and YOLO v3 on Hand-drawn Images" (2019). University Honors Theses. Paper 693

[11] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and Geodesic-Bound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[12] НПО Регион. Режим доступа-URL: https://nporegion.ru/laboratorii/

[13] Sichkar V.N., Kolyubin S.A. Real time detection and classification of traffic signs based on YOLO version 3 algorithm. Scientific and Technical Journal of Information Technologies, Mechanics and Optics, 2020, vol. 20, no. 3, pp. 418–424 (in English). doi: 10.17586/2226-1494-2020-20-3-418-424.

[14] M. Everingham, L. Van Gool, Williams, C.K.I. et al. “The PASCAL Visual Object Classes (VOC) Challenge”, International Journal of Computer Vision, 2010, vol. 88, pp. 303–338.

[15] Д.Е, Иванов & Полевой, Дмитрий & Sholomov, Dmitry. (2018). Отбор информативных элементов для обучения легкого сверточного нейросетевого классификатора в условиях сильного дисбаланса обучающей выборки. 199-204. 10.14357/20790279180523.

[16] Ярышев С.Н., Рыжова В.А., Технологии глубокого обучения и нейронных сетей в задачах видеоанализа – СПб: Университет ИТМО, 2022 – 82 с.

[17] Zhou, Shuren et al. “Improved VGG Model for Road Traffic Sign Recognition.” *Cmc-computers Materials & Continua* 57 (2018): 11-24.

[18] Possatti, Lucas C. et al. “Traffic Light Recognition Using Deep Learning and Prior Maps for Autonomous Cars”, 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-8.

J. Redmon, A. Farhadi. “YOLOv3: An Incremental Improvement”, ArXiv abs/1804.02767, 2018, pp. 1

Разработка стратегии торговли биткоином с использованием методов машинного обучения

Личко Д.А.
кафедра инженерной кибернетики
НИТУ МИСИС
Москва, Россия
m1902984@edu.misis.ru

***Аннотация* — Криптовалюты стали неотъемлемой частью развивающейся индустрии цифровых активов. Однако, несмотря на их популярность, рынок криптовалют все ещё остается малоисследованным в сравнении с традиционными финансовыми активами. В данном исследовании был проведен сравнительный анализ различных моделей машинного обучения, примененных к почасовым данным цены биткоина, с целью предсказания ее тренда (рост или падение). В ходе исследования были рассмотрены модели, основанные на реккурентных и свёрточных нейронных сетях, а также модели градиентного бустинга. Целью исследования является выявление эффективных моделей, способных предсказывать динамику цены биткоина и формирование на основе обученной модели торговой стратегии. Результаты работы могут быть полезны для инвесторов, трейдеров и исследователей, стремящихся понять и использовать потенциал криптовалют в современной экономике.**

Ключевые слова — криптовалюта, машинное обучение, нейронные сети, градиентный бустинг, биткоин, LSTM, GRU, MLP, CatBoost, классификация временных рядов, инвестирование, технический анализ

I. Введение

Криптовалюты являются значимой частью новой и бурно развивающейся индустрии цифровых активов, которая в последние пару лет стала очень популярна в мире. Уже в 2021 году капитализация рынка криптовалют перевалила за 3 триллиона долларов. К тому же технология блокчейна является одним из главных столпов новой децентрализованной концепции интернета – Web3. Все эти факты указывают на то, что криптовалюты заняли устойчивое положение на современном рынке и являются перспективным направлением исследований (количество научных работ растет с каждым годом [1]), средством для инвестирования.

Отличительной особенностью рынка криптовалют по сравнению с финансовым является ограниченная возможность фундаментального анализа. В то время как при анализе акций необходимо принимать во внимание финансовые и производственные показатели деятельности компании, отчеты и дивиденды, в случае с криптовалютами мы можем опираться только на данные торгов и данные блокчейна. Поэтому можно выдвинуть гипотезу, что для определения автоматической стратегии выгодной торговли криптовалютой достаточно этих данных.

Для автоматического составления торговой стратегии могут применяться методы машинного обучения. В последние годы их успешно применяют в различных областях: от распознавания изображений [2] и пилотирования [3-4] до работы с документами [5-6].

II. Анализ источников

На основе обзора источников с 2014 по 2022 год больше 50% научных статей, посвященных прогнозированию тренда криптовалюты решали задачу регрессии, в то время как задача классификации рассматривалась в 35% статей [7]. Однако существующие решения задачи регрессии будущих цен криптовалюты не имеют достаточную точность для использования в торговой стратегии [8], к тому же в процессе определения стратегии торговли важно именно направление изменения цены, а не ее абсолютное значение. Поэтому в качестве математической задачи в данной работе принята бинарная классификация.

Исследования [9-11] использовали различные методы классификации дневного тренда биткоина. В качестве входных данных были приняты исторические данные цены (OHLCV):

- В исследовании [9] были протестированы рекуррентные нейронные сети и модель ARIMA. Наилучший результат был у модели LSTM (accuracy = 0.52);
- В рамках работы [10] были обучены CNN-LSTM модели и применены методы бэггинга. Благодаря этому получилось достичь значения accuracy на тестовой выборке в 0.5466;
- Исследование [11] использовало метод главных компонент для улучшения точности моделей. Самая точная модель – XGBoost (метод градиентного бустинга). Благодаря использованию метода главных компонент, accuracy этой модели выросло с 0.49 до 0.54.

В свою очередь исследования, в которых были добавлены дополнительные признаки в датасет для обучения, показывают более высокую точность классификации. Так, исследование [12] в качестве признаков использовало сгенерированные на основе истории цены технические индикаторы. Были использованы несколько моделей, в т. ч. логистическая регрессия, MLP, случайный лес, метод опорных векторов. Случайный лес был определен как лучшая модель для бинарной классификации дневных данных со значением accuracy = 0.62.

Автор статьи [13] в качестве признаков использовал технические индикаторы, значения популярности

интернет запросов через Google trends и рыночные фундаментальные показатели. Обученная на этих данных модель XGBoost улучшила метрику accuracy относительно статьи, в которой использовались только технические индикаторы.

В статье [14] решалась задача бинарной классификации тренда Биткоина с гранулярностью в час. Для обучения были использованы только исторические данные цены криптовалюты. Точность обученных моделей (Логистическая регрессия, SVM, RNN) получилась ниже 50%, что может говорить о неудачном подходе к решению задачи и необходимости использования дополнительных признаков для обучения. Только модель ARIMA показала сравнительно неплохую точность в 53%.

Авторы статьи [15] тоже использовали часовые данные, но также были сгенерированы технические индикаторы. Благодаря этому была достигнута точность классификации примерно в 54% для моделей MLP, LST и MALSTM-FCN. Сверточная нейронная сеть в свою очередь показала плохой результат в 44%.

III. Данные

A. *Описание данных*

В рамках данного исследования используются часовые данные криптовалюты Биткоин. Для загрузки данных был написан скрипт, работающий с API криптовалютной биржи Binance. С его помощью были выгружены почасовые исторические данные цен биткоина (OHLC - Open, High, Low, Close) и объема обращения (Volume) за период с 17.08.2017 по 15.10.2023.

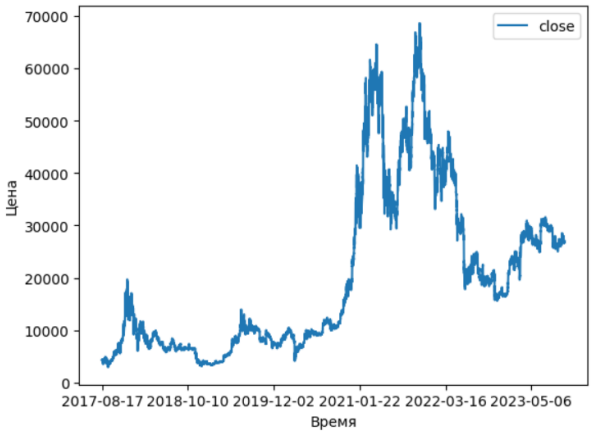


Рис. 1. График цены биткоина за период с 17.08.2017 по 15.10.2023

При анализе данных было обнаружено, что признак Volume в датасете имеет нулевые значения. Было выяснено, что несколько раз за взятый период биржа была недоступна из-за ошибок, поэтому в это время объем обращения падал до нуля. Нулевые значения были заполнены ближайшим ненулевым значением.

Анализ источников показал, что OHLCV данных недостаточно для точной классификации курса Биткоина. Поэтому на основе загруженных данных были рассчитаны технические индикаторы, представленные приложении A.

Итоговый датасет представляет из себя таблицу, в которой каждая строка описывает состояние биткоина на один час, а каждый столбец – признак. Всего датасет имеет 53844 строк и 100 столбцов.

IV. Проведенные исследования и результаты

A. *Предобработка датасета*

Для того, чтобы обучить нейронные сети на данных, которые являются временным рядом, необходимо преобразовать датасет в определенный формат. Для этого был использован метод скользящего окна:

Пусть s - размер окна, t - временная метка, X - исходный датасет, Y - датасет после применения метода скользящего окна. Тогда датасет преобразуется согласно формуле (1).

$$Y_t = X_{t-s..t} \quad (1)$$

Большинство признаков в получившимся датасете не нуждались в нормализации, так как технические индикаторы зачастую находятся в границах [0; 100] или [-100; 100]. Однако признаки OHCLV и такие индикаторы, как например SMA, EMA, MACD представлены в абсолютных величинах. Для нормализации этих признаков был использован метод нормализации первым элементом скользящего окна, описанный в статье [8]. С нормализацией метод скользящего окна описывается по формуле (2).

$$Y_t = X_{t-s..t}/X_{t-s} \quad (2)$$

В датасете была рассчитана целевая переменная. Если цена Биткоина на следующий день растет, целевая переменная = 1, в обратном случае = 0.

Датасет был разделен на обучающую и тестовую выборки в соотношении 80%-20%. Разделение осуществлялось без перемешивания сэмплов, так как при составлении торговой стратегии критически важно иметь хорошую точность предсказания тренда именно на наиболее актуальных данных.

B. *Выбор метрик*

Для оценивания точности бинарной классификации были взяты метрики accuracy и F1. Accuracy - доля правильно классифицированных объектов. F1 - метрика, рассчитываемая на основе значений матрицы ошибок, является гармоническим средним между Precision и Recall. Где Precision - процент объектов положительного класса, который правильно классифицировали, относительно всех классификаций этого класса, а Recall - доля объектов положительного класса, которые были правильно классифицированы, относительно всех объектов класса. В частности, в данной работе используется взвешенный F1, который более информативен при неравномерных классах.

C. *Использованные модели машинного обучения*

InceptionTime [16] - это архитектура нейронной сети, специально разработанная для обработки временных рядов. Она использует одномерные свертки для анализа временных данных. Архитектура состоит из нескольких блоков, каждый из которых содержит несколько Inception модулей. Inception модуль представляет собой комбинацию различных фильтров и операций сжатия и

объединения, что позволяет сети извлекать различные уровни абстракции из временных данных.

LSTM [17] - архитектура рекуррентной нейронной сети, которая хорошо работает с последовательными данных. LSTM использует ячейки памяти для хранения информации о предыдущих состояниях и принимает решения о том, какую информацию сохранить и какую забыть.

GRU [18] - упрощенная реализация рекуррентной сети. Имеет меньше количество gates, чем LSTM. Благодаря этому быстрее обучается и требует меньше данных для обучения.

XGBoost [19] - это градиентный бустинговый алгоритм, который широко используется в задачах регрессии и классификации. Он основан на ансамбле деревьев решений и использует градиентный спуск для построения модели. Реализовано с помощью библиотеки XGBoost.

CatBoost [20] - другая реализация алгоритма градиентного бустинга с дополнительными функциями обработки категориальных признаков, выбросов и пропущенных значений. Реализовано с помощью библиотеки CatBoost.

RandomForest - это ансамблевый метод машинного обучения (бэггинг), который состоит из множества деревьев решений. Каждое дерево строится независимо на случайной подвыборке данных и случайном подмножестве признаков. Затем, при прогнозировании, каждое дерево дает свой прогноз, и итоговый прогноз определяется голосованием или усреднением по всем деревьям. Реализовано с помощью библиотеки scikit-learn.

ROCKET (RandOm Convolutional KErnel Transform) - это метод обработки временных рядов, описанный в статье 2019 года [21]. Он основан на применении случайных сверток. ROCKET использует набор случайных фильтров для извлечения различных признаков из временных рядов. Затем полученные признаки подаются на вход линейному классификатору. Он особенно полезен в задачах классификации временных рядов.

Все представленные архитектуры нейронных сетей и ROCKET были реализованы с помощью библиотеки tsai [22].

D. Оптимизация гиперпараметров

Для выбора оптимальных признаков был произведен корреляционный анализ и использован метод рекуррентного отбора признаков.

Для оптимизации гиперпараметров моделей использована Баесовская оптимизация. Этот метод требует меньше итераций, чем поиск по сетке или случайный поиск за счет моделирования функции оценки производительности модели. Реализация данного метода взята из Python библиотеки optuna [23]. Ниже представлены оптимизированные гиперпараметры для каждой модели:

InceptionTime:

- window: 6
- batch: 413
- nf: 32
- ks: 40

LSTM:

- hidden_size: 101
- n_layers: 3;
- rnn_dropout: 0.1;
- bidirectional: True;
- fc_dropout: 0.3.

GRU:

- hidden_size: 139
- n_layers: 2;
- rnn_dropout: 0.2;
- bidirectional: True;
- fc_dropout: 0.1.

XGBoost:

- lambda: 0.8310197824164319;
- alpha: 0.1636404767005528;
- colsample_bytree: 0.4;
- subsample: 0.4;
- learning_rate: 0.008;
- n_estimators: 1883;
- max_depth: 12;
- min_child_weight: 65.

CatBoost:

- iterations: 1141;
- learning_rate: 0.021756640134175127;
- depth: 6;
- l2_leaf_reg: 12;
- boosting_type: 'Ordered';
- bootstrap_type: 'Bayesian';
- random_strength: 0.23380846423713583;
- od_type: 'IncToDec';
- od_wait: 27;
- bagging_temperature: 4.

RandomForest:

- n_estimators: 486;

- max_depth: 5;
- min_samples_split: 23;
- min_samples_leaf: 9.

ROCKET:

- n_kernels: 10_000;
- kss: [7, 9, 11].

E. Результаты обучения

В таблице 1 представлены результаты обучения моделей.

ТАБЛИЦА I. РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Модель	Accuracy	F1_w
InceptionTime	0.5396	0.5389
LSTM	0.5392	0.5392
GRU	0.5378	0.5374
XGBoost	0.5670	0.5664
CatBoost	**0.5713**	**0.5692**
RandomForest	0.5687	0.5658
ROCKET	0.5288	0.5267

По результатам обучения видно, что наиболее точная модель - CatBoost. Ее точность на порядок превосходит модели из исследований, использующих часовые данные [14-15]. В целом методы классического машинного обучения с задачей справились лучше, чем нейронные сети. Среди нейронных сетей лучшая модель по accuracy – InceptionTime, а по F1 - LSTM.

На рис. 2 представлена матрица ошибок на тестовой выборке для CatBoost.

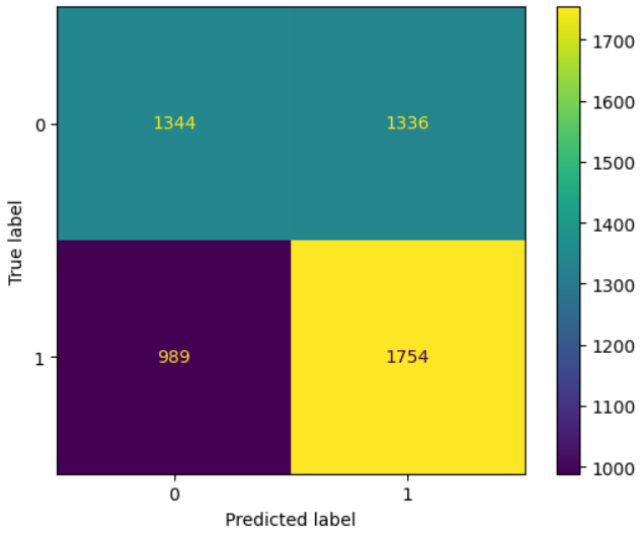


Рис. 2. Матрица ошибок на тестовой выборке для модели CatBoost

F. Результаты тестирования торговой стратегии

Для того, чтобы понимать, насколько обученная модель применима для использования в реальной торговле, недостаточно только метрик классификации. Необходимо смоделировать торговую стратегию, используя период исторических данных, не использовавшийся при обучении. Для этого тестирования был выбран промежуток с 01.10.2022 по 13.10.2023 (всего 9048 временных меток). Тестирование осуществлялось с помощью библиотеки backtest.py [24].

В рамках тестирования для каждой даты в периоде было получено предсказание модели и на его основе открыта или закрыта позиция (при классификации классом 1, если позиция еще не была открыта – она открывалась, а при классификации классом 0 - закрывалась). Таким образом, пройдя от начала периода тестирования до конца, были смоделированы реальные торги. Полученные результаты были сравнены со стратегией Buy&Hold, это отражено в таблице 2.

ТАБЛИЦА II. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ ТОРГОВОЙ СТРАТЕГИ

Торговая стратегия	Выгода	Процент успешных транзакций
CatBoost модель	29.2%	72.8%
Buy&Hold	13.7%	-

По результатам тестирования можно говорить об успешности применения торговой стратегии. Торговля Биткоином с ее помощью принесла прибыль почти в 30% за год инвестирования, в то время как простая покупка и удержание криптовалюты принесли бы прибыль в 14%. Также результаты торговой стратегии превосходят результаты, представленные в статье [8] - там за полтора года инвестирования, прибыль составила примерно 9 процентов.

На рис. 3 представлен график изменения стоимости актива при торговле с помощью описанной торговой стратегии. Красная точка обозначает наибольшую локальную просадку, голубая - наибольшее повышение стоимости, а синяя - стоимость в конце периода.

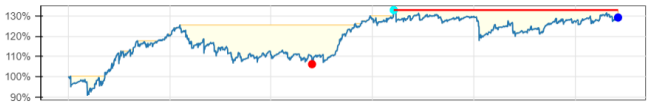


Рис 3. График изменения стоимости актива

V. ЗАКЛЮЧЕНИЕ

В рамках данной работы был проведен анализ литературы, посвященной предсказанию тренда криптовалюты Биткоин. На его основании были определены наиболее подходящие исходные данные, признаки и модели машинного обучения.

Были загружены исторические данные Биткоина и сгенерировано большое количество технических индикаторов. Получившиеся данные были проанализированы, обработаны и сформированы в датасет.

Было обучено несколько моделей машинного обучения, для улучшения их качества была произведена оптимизация гиперпараметров. Наилучшее качество в 0.57 показала модель CatBoost. Среди нейронных сетей лучшее качество показала InceptionTime - 0.54.

На основе CatBoost модели была сформирована торговая стратегия. Она была протестирована на данных за год и показала прибыль почти в 30%. На основании этих данных можно говорить о применимости машинного обучения в торговле криптовалютой.

В дальнейшем планируется разработать торгового бота для автоматической торговли в соответствии с предложенной торговой стратегией. Также планируется повышать качество модели за счет добавления новых признаков, например, основанных на сентимент анализе социальных сетей и на фундаментальных показателях криптовалюты.

Литература

[1] Fang, F., Ventre, C., Basios, M. et al. Cryptocurrency trading: a comprehensive survey // Financial Innovation. – 2022. – V. 8. – N 13.

[2] Kudryashov A. A., Mishchanin M. A., Sadekov R. N. Food recognition using deep learning networks and order history for smart canteen checkout automation.

[3] Ali, B., Sadekov, R.N., & Tsodokova, V.V. (2022). A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems. Gyroscopy and Navigation, 13, 241-252.

[4] R. R. Bikmaev, A. A. Polukarov and R. N. Sadekov, "Visual Localization of a Ground Vehicle Using a Monocamera and GeodesicBound Road Signs," 2020 27th Saint Petersburg International Conference on Integrated Navigation Systems (ICINS), St. Petersburg, Russia, 2020, pp. 1-5, doi: 10.23919/ICINS43215.2020.9133769.

[5] Ilin, D., Novikov, D., Polevoy, D.V., & Nikolaev, D.P. (2018). Fast words boundaries localization in text fields for low quality document images. International Conference on Machine Vision.

[6] Arlazarov, V.L., Arlazarov, V.V., Bulatov, K.B., Chernov, T.S., Nikolaev, D.P., Polevoy, D., Sheshkus, A.V., Skoryukina, N.S., Slavin, O.A., & Usilin, S.A. (2022). Mobile ID Document Recognition–Coarse-to-Fine Approach. Pattern Recognition and Image Analysis 32, 89-108.

[7] «Cryptocurrency market trend and direction prediction using Machine Learning: A Comprehensive Survey» M. Yamin, M. Chaudhry // Authorea. URL: https://www.authorea.com/doi/full/10.22541/au.167285886.66422340 (доступ 20.10.2023)

[8] Ji S., Kim J., Im H. A Comparative Study of Bitcoin Price Prediction Using Deep Learning. // Mathematics. – 2019. – V. 7. – N. 10.

[9] McNally et. al. Predicting the Price of Bitcoin Using Machine Learning. // 2018 26th Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP). – 2018. – V. 26.

[10] Livieris I. et al. Ensemble Deep Learning Models for Forecasting Cryptocurrency Time-Series. // Algorithms. – 2020. – V. 13. – N. 5.

[11] «Signal Prediction in Cryptocurrency Tradeoperations: A Machine LearningBased Approach» J. Toledo, D. Souza // SSRN. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=4062476 (доступ 20.10.2023)

[12] Akyildirim E., Goncu A., Sensoy A. Prediction of cryptocurrency returns using machine learning. // Annals of Operations Research. – 2021. – V. 297.

[13] Shchetinin E. On methods of building the trading strategiesin the cryptocurrency markets. // Discrete & Continuous Models & Applied Computational Science. – 2022. –V. 30. – N. 1. – P. 79–87.

[14] Mangla N., Bitcoin Price Prediction Using Machine Learning. // INTERNATIONAL JOURNAL OF INFORMATION AND COMPUTING SCIENCE. – 2019. –V. 6. – N. 5.

[15] Ortu M., Uras N. On Technical Trading and Social Media Indicators in Cryptocurrencies' Price Classification Through Deep Learning // Expert Systems with Applications. – 2022. –V. 198.

[16] Ismail Fawaz H., Lucas B., Forestier G. et al. InceptionTime: Finding AlexNet for time series classification // Data Mining and Knowledge Discovery. – 2020. –V. 34.

[17] Hochreiter, S., Schmidhuber, J. Long Short-Term Memory. // Neural Computation. – 1997. – V. 9. – P. 1735–1780.

[18] Junyoung C, et al. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. // arXiv:1412.3555. – 2014.

[19] Chen T., Guestrin C. XGBoost: A Scalable Tree Boosting System. // The 22ndACM SIGKDD International Conference. – 2016. – P. 785–794

[20] Prokhorenkova L., Gusev G., Vorobev A., Dorogush A., Gulin A. CatBoost: unbiased boosting with categorical features. // The 32nd International Conference on Neural Information Processing Systems. – 2018. – P. 6639–6649.

[21] Dempster A., Petitjean F., Webb G.I. ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels // Data Mining and Knowledge Discovery. – 2018. – V. 34. – P. 1454–1495.

[22] tsai - A state-of-the-art deep learning library for time series and sequential data // Github. URL: https://github.com/timeseriesAI/tsai (доступ 20.10.2023)

[23] Takuya A. et al. Optuna: A Next-generation Hyperparameter Optimization Framework // 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. – 2019. – P. 2623–2631 .

[24] Backtest trading strategies with Python // Github. URL: https://github.com/kernc/backtesting.py (доступ 20.10.2023)

ПРИЛОЖЕНИЕ А

Формулы расчёта технических индикаторов.

Пусть O_t – цена начала часа t, H_t – наибольшая цена криптовалюты за час t, L_t – наименьшая цена криптовалюты за час t, C_t – цена закрытия часа t, V_t – объем торгов за час t, N – длина окна. Тогда формулы технических индикаторов представлены в таблице 3.

ТАБЛИЦА III. ТЕХНИЧЕСКИЕ ИНДИКАТОРЫ

Индикатор	**Формула**
Простое скользящее среднее	$$SMA_t(N) = \frac{1}{N}\sum_{i=0}^{N-1} P_{t-i}$$
Экспоненциальное скользящее среднее	$$\alpha = \frac{2}{N\ +\ 1}$$ $$EMA_t(N) = \alpha * P_t + (1-\alpha) * EMA_{t-1}$$
Взвешенное скользящее среднее	$$WMA_t(N) = \sum_{i=0}^{N-1} P_{t-i}\ * \frac{2*(N-i)}{(1+N)*N}$$
MACD индикатор	$$MACD_t = EMA_t(26) - EMA_t(12)$$
MACD сигнал	$$MACD\_SIGNAL_t = EMA_t(9)$$
MACD гистограмма	$$MACD\_HIST_t = MACD_t - MACD\_SIGNAL_t$$
Williams %R	$$HH_t(N) = H_{t-i}$$ $$LL_t(N) = L_{t-i}$$ $$WR_t(N) = \frac{HH_t(N)\ -\ C_t}{HH_t(N)\ -\ LL_t(N)} * 100$$
Psychological line	$$signum(x) = \begin{cases} 1, & x >= 0 \\ 0, & x < 0 \end{cases}$$ $$PSY_t(N) = \frac{1}{N}\sum_{i=0}^{N-1} signum(C_{t-i} - C_{t-i-1})$$
DIFF	$$DIFF_t = C_t - C_{t-1}$$
Relative Strength Index	$$GAIN_t(N) = \frac{1}{N}\sum_{i=0}^{N-1} signum(DIFF_{t-i}) * DIFF_{t-i}$$ $$LOSS_t(N) = \frac{1}{N}\sum_{i=0}^{N-1} signum(-1 * DIFF_{t-i}) * \ \lvert DIFF_{t-i}\rvert$$ $$RSI_t(N) = 100 - \frac{100}{1\ +\ \frac{GAIN_t(N)}{LOSS_t(N)}}$$

Индикатор	Формула
Money Flow Index	$$TYPICAL_t = \frac{H_t + L_t + C_t}{3}$$ $$DIFF_t = TYPICAL_t - TYPICAL_{t-1}$$ $$GAIN_t(N) = \sum_{i=0}^{N-1} signum(DIFF_{t-i}) * TYPICAL_t * V_t$$ $$LOSS_t(N) = \sum_{i=0}^{N-1} signum(-1 * DIFF_{t-i}) * TYPICAL_t * V_t$$ $$MFI_t(N) = 100 - \frac{100}{1 + \frac{GAIN_t(N)}{LOSS_t(N)}}$$
Directional Movement Index	$$UP_t = H_t - H_{t-1}$$ $$DOWN_t = L_{t-1} - L_t$$ $$POS_t = signum(UP_t - DOWN_t) * UP_t$$ $$NEG_t = signum(DOWN_t - UP_t) * DOWN_t$$ $DMP_t(N) = EMA_t(N)$ по значениям POS $DMN_t(N) = EMA_t(N)$ по значениям NEG
Bollinger band	$$S_t = \sqrt{\frac{1}{N}\sum_{i=1}^{N-1}(C_t - SMA_t(N))^2}$$ $$BBUP_t(N) = SMA_t(N) + 2 * S_t$$ $$BBDOWN_t(N) = SMA_t(N) - 2 * S_t$$
Keltner channel indicator	$$KCUP_t(N) = EMA_t(N) + 2 * max(H_t - L_t, H_t - C_{t-1}, C_{t-1} - L_t)$$ $$KCDOWN_t(N) = EMA_t(N) - 2 * max(H_t - L_t, H_t - C_{t-1}, C_{t-1} - L_t)$$
TTM Squeeze	$SQZON_t(\mathrm{T}) = \begin{cases} 1, BBDOWN_t(N) > KCDOWN_t(N) \text{ и} BBHIGH_t(N) < KCHIGH_t(N) \\ 0, BBDOWN_t(N) \le KCDOWN_t(N) \text{ или } BBHIGH_t(N) \ge KCHIGH_t(N) \end{cases}$ $SQZOFF_t(\mathrm{T}) = \begin{cases} 1, BBDOWN_t(N) < KCDOWN_t(N) \text{ и } BBHIGH_t(N) > KCHIGH_t(N) \\ 0, BBDOWN_t(N) \ge KCDOWN_t(N) \text{ или } BBHIGH_t(N) \le KCHIGH_t(N) \end{cases}$
Schaff Trend Cycle	$$LMACD_t(N) = MACD_{t-i}$$ $$HMACD_t(N) = MACD_{t-i}$$ $$STC_t(N) = \begin{cases} \frac{MACD_t - LMACD_t(N)}{HMACD_t(N) - LMACD_t(N)}, LMACD_t(N) > 0 \\ STC_{t-1}(N), LMACD_t(N) \le 0 \end{cases}$$

Обнаружение ветрогенераторов при помощи компьютерного зрения

М. А. Коновалов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m1805775@edu.misis.ru

***Аннотация* — с увеличением доли возобновляемых источников энергии в мире, ветроэнергетика становится все более значимым фактором. Эффективное обнаружение и мониторинг состояния ветрогенераторов является критическим элементом для обеспечения их устойчивой работы и максимизации производительности. В данной работе исследуется применение компьютерного зрения для обнаружения ветрогенераторов. Ветрогенераторы — это устройства, используемые для преобразования энергии ветра в электрическую энергию. Цель проведения данного исследования – анализ эффективности алгоритмов и моделей компьютерного зрения, которые смогут точно и надежно обнаруживать ветрогенераторы на различных типах изображений и видео. В данной работе рассматривается решение с открытым исходным кодом, а в частности архитектура - YOLOv8. Для проведения анализа результатов (метрик), в качестве входных данных используются изображения, полученные с камер БПЛА: WTIDFCV и WTD.**

Ключевые слова: компьютерное зрение, обнаружение объектов, распознавание объектов, ветрогенераторы, YOLOv8, Box Loss, mAP.

I. Введение

Ветрогенераторы являются неотъемлемой частью производства возобновляемой энергии, но их техническое обслуживание и мониторинг представляют собой сложную задачу, особенно для крупных ветроферм. Традиционная ручная инспекция занимает много времени, требует значительных затрат и может быть опасной. С другой стороны, беспилотные летательные аппараты (БПЛА) предлагают эффективную и безопасную альтернативу для инспекции ветрогенераторов [1]. Однако анализ данных, собранных БПЛА, может потребовать много времени и требует наличия квалифицированных специалистов. Одно из возможных решений - использовать обнаружение объектов для автоматизации процесса анализа и выявления дефектов ветрогенераторов.

Несмотря на потенциальные преимущества использования БПЛА для инспекции ветрогенераторов, текущий процесс инспекции может быть неэффективным и затратным. Требуется наличие квалифицированных специалистов для ручного анализа данных, собранных БПЛА, и это может стать узким местом в процессе инспекции. Кроме того, ручная инспекция может быть опасной и дорогостоящей, особенно в условиях неблагоприятных погодных условий или сложного рельефа местности. Традиционный процесс инспекции также имеет ограниченную область охвата и может упустить дефекты, которые не видны с уровня земли. Эти проблемы могут привести к снижению энергетической производительности и увеличению затрат на обслуживание.

Обнаружение объектов может автоматизировать процесс инспекции ветрогенераторов и предоставить эффективное и безопасное решение для мониторинга и обслуживания ветроферм. Алгоритмы обнаружения объектов, анализируя данные, собранные БПЛА, могут выявлять дефекты ветрогенераторов, такие как трещины, коррозия и другие виды повреждений. Это может помочь обнаружить и диагностировать проблемы до их усугубления и повысить эффективность обслуживания ветрогенераторов. Кроме того, обнаружение объектов может охватить большую площадь и предоставить комплексное представление обо всей ветроферме. Эта технология имеет потенциал для улучшения безопасности и надежности ветрогенераторов и способствования внедрению возобновляемой энергии.

Методы глубокого обучения демонстрируют высокую производительность и способность к обобщению во многих областях и типах задач [2, 3, 4], включая классификацию и обнаружение объектов. В частности, детекторы объектов общего назначения тщательно исследованы и разработаны для решения задач, связанных с обнаружением объектов на изображениях [5, 6].

Один из основных современных детекторов, YOLO (You Only Look Once)[7], заслуживает особого внимания. В частности, YOLOv8 является передовым алгоритмом обнаружения объектов, который широко применяется для обнаружения объектов в режиме реального времени. Благодаря своей эффективности и популярности, YOLOv8 стал предпочтительным выбором для точной идентификации и определения местоположения объектов на изображениях.

II. Наборы данных

Для обучения и тестирования рассматриваемой в данной работе нейросети использовались некоторые наборы данных. Рассмотрим используемые открытые наборы.

A. WTIDFCV

Этот набор данных представляет собой комплексную коллекцию изображений ветряных турбин, снятых на динамичном и меняющемся фоне. Этот набор данных разработан специально для дронов и заинтересованы в изучении и анализе ветроэнергетических установок.

Каждое изображение в наборе данных тщательно отобрано и содержит детальные снимки ветряных турбин различных моделей, размещенных в разных местоположениях и под разными углами.

Для увеличения количества данных используется аугментация [8]:

- 50% вероятность горизонтального переворота

- Случайное размытие по Гауссу от 0 до 3 пикселей.
- Случайная регулировка экспозиции от -25% до +25%.
- Ограничительная рамка: Шум: до 5% пикселей.

Предварительное разделение: 87% обучение, 9% валидация, 4% тестирование (2885 изображений).

Рис. 1. Примеры кадров при дневном ясном освещении

Рис. 2. Примеры кадров при дневном туманном освещении

Рис. 3. Примеры кадров при ослепляющем освещении

B. WTD

Набор данных WTD содержит 1700 ручных аннотаций кадров. WTD является аналогом датасета *A*.

Для увеличения количества данных используется аугментация [8]:

- Случайное вращение на 1°.
- Случайная регулировка яркости от - 3% до + 3%.
- Случайная регулировка экспозиции от -10% до +10%.

Рис. 4. Примеры кадров наслаивания

III. Нейросетевая архитектура

YOLOv8

Основное отличие YOLO [7] от других алгоритмов сверточной нейронной сети (CNN) [9, 10], используемых для обнаружения объектов, заключается в том, что он очень быстро опознает объекты в режиме реального времени. Принцип работы YOLO подразумевает ввод сразу всего изображения, которое проходит через сверточную нейронную сеть только один раз. Именно поэтому он называется “Стоит только раз взглянуть”. В других алгоритмах этот процесс происходит многократно, то есть изображение проходит через CNN снова и снова. Так что YOLO обладает преимуществом высокоскоростного обнаружения объектов, чем не могут похвастать другие алгоритмы.

Проблема обнаружения объектов более сложна, чем задача классификации [3], которая также может распознавать объекты, но не указывает, где объект находится на изображении. Кроме того, классификация не работает на изображениях, содержащих более одного объекта. Алгоритм YOLO применяет один прямой проход по сети к полному изображению, а затем делит изображение на области и прогнозирует ограничивающие рамки и вероятности для каждой области. Эти ограничивающие рамки взвешиваются предсказанными вероятностями. После non-max подавления (гарантирующего, что алгоритм обнаружения объектов обнаруживает каждый объект только один раз) он затем выводит распознанные объекты вместе с ограничивающими рамками.

С помощью YOLO одна CNN одновременно прогнозирует несколько ограничивающих рамок и вероятности

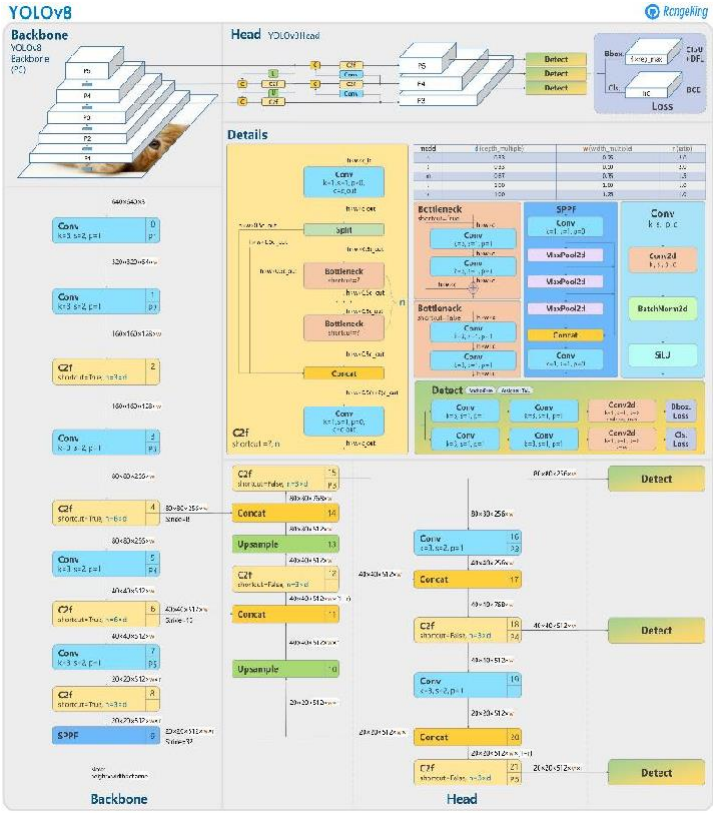


Рис. 5. Архитектура YOLOv8

классов для этих рамок. YOLO обучается на полных изображениях и напрямую оптимизирует производительность обнаружения. Эта модель имеет ряд преимуществ перед другими методами обнаружения объектов.

Архитектура CNN для YOLO была вдохновлена моделью GoogLeNet [11] для классификации изображений.

В данной работе модель YOLOv8 была обучена для детектирования и распознавания ветрогенераторов. Нейронная сеть на выходе имеет два класса – «turbine» и «background».

Открытые наборы данных упомянутые выше были использованы для обучения, валидации и тестирования нейронной сети.

Архитектура YOLOv8 (рис. 5) является одной из версий популярной семейства алгоритмов для обнаружения объектов в реальном времени. Ниже представлен обзор основных черт архитектуры YOLOv8:

Основные принципы:

- Однопроходный подход: YOLO в своей основе оперирует принципом однопроходного обнаружения, что означает, что изображение анализируется за один проход, а не в несколько этапов, как в некоторых других алгоритмах.
- Детекция на уровне якорей (Anchor-based detection): YOLO использует предопределенные якори для улучшения точности и стабильности обнаружения объектов различных размеров.

Архитектурные улучшения в YOLOv8:

- Нейронная сеть CSPDarknet53 [12]: В YOLOv8 применяется CSPDarknet53 в качестве базовой нейронной сети. Это модифицированная версия Darknet, которая включает в себя блок "cross-stage" (CSP), что способствует улучшению эффективности и обобщающей способности модели.
- Пирамидальная сеть (PANet [13]): используется PANet для объединения различных уровней признаков, что помогает лучше обрабатывать объекты различных масштабов.
- SPP (Spatial Pyramid Pooling) [14]: В YOLOv8 применяется SPP для увеличения размера поля зрения и улучшения обнаружения мелких объектов.

Конфигурации:

- YOLOv8 имеет различные конфигурации, обозначаемые как YOLOv8-S, YOLOv8-M, YOLOv8-L, и YOLOv8-XL. Каждая из этих конфигураций имеет разные архитектурные характеристики, такие как количество слоев и параметров, что позволяет выбирать модель в зависимости от требований к производительности и точности.

Обучение:

- Обучение проводится на больших наборах данных, таких как COCO (Common Objects in Context) [15], для достижения высокой обобщающей способности.
- Используется метод обучения с учителем (supervised learning) с использованием размеченных данных, где модель обучается определять классы и ограничивающие рамки объектов.

IV. Анализ результатов

Анализ результатов производительности детектирования ветрогенераторов включает в себя оценку точности, полноты обнаружения и других метрик для измерения эффективности модели [16].

Производительность модели:

- Train Box Loss**:** измеряет разницу между предсказанными ограничивающими рамками и фактическими ограничивающими рамками объектов в обучающих данных. Меньшая потеря области означает, что предсказанные моделью ограничивающие рамки более точно соответствуют фактическим ограничивающим рамкам.

- Train Class Loss**:** потери класса при обучении измеряет разницу между предсказанными вероятностями класса и фактическими метками класса объектов в обучающих данных. Меньшая потеря класса означает, что предсказанные моделью вероятности класса более точно соответствуют фактическим меткам класса.

- Train DFL Loss: потери DFL (динамического обучения признаков) измеряет разницу между предсказанными картами признаков и фактическими картами признаков объектов в обучающих данных. Меньшая потеря DFL означает, что

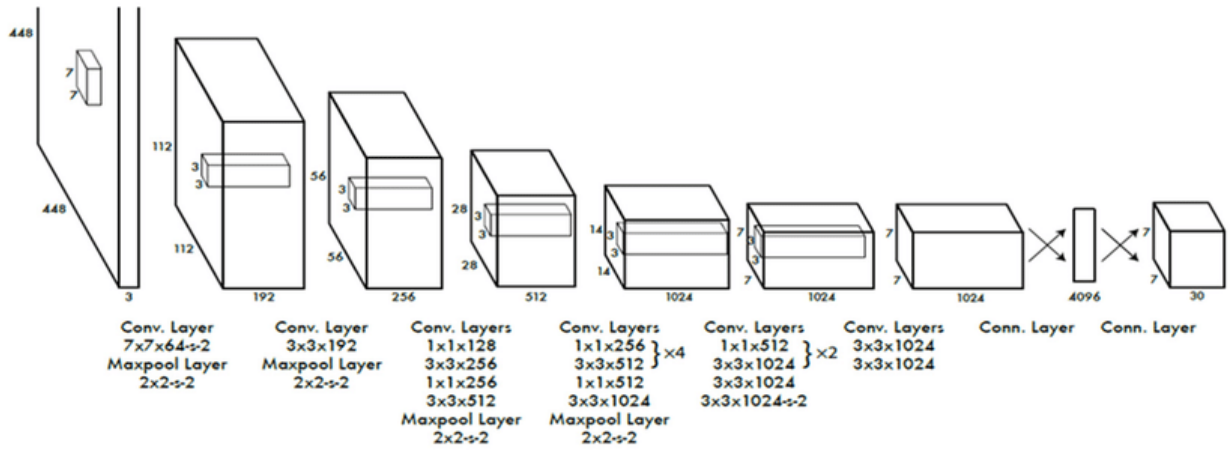


Рис. 6. Архитектура CNN для YOLO

предсказанные моделью карты признаков более точно соответствуют фактическим картам признаков.

- Metrics Precision (B): измеряет долю верно обнаруженных областей среди всех предсказанных ограничивающих рамок. Более высокая точность означает, что модель лучше определяет верно обнаруженные области и минимизирует ложноположительные результаты.

$$Precision = \frac{TP}{TP + FP}$$

- Metrics Recall (B): измеряет долю верно обнаруженных областей среди всех фактических ограничивающих рамок. Более высокая полнота означает, что модель лучше определяет все верно обнаруженные области и минимизирует ложноотрицательные результаты.

$$Recall = \frac{TP}{TP + FN}$$

- Metrics mAP50 (B): измеряет среднюю совокупную точность модели по разным категориям объектов с порогом пересечения-объединения (IoU) 50%. Более высокое значение mAP50 означает, что модель лучше идентифицирует и локализует объекты разных категорий.

- Metrics mAP50-95 (B): измеряет среднюю совокупную точность модели по разным категориям объектов с порогами IoU от 50% до 95%. Более высокое значение mAP50-95 означает, что модель лучше идентифицирует и локализует объекты разных категорий с широким диапазоном порогов IoU.

Рис. 7. Метрики при обучении

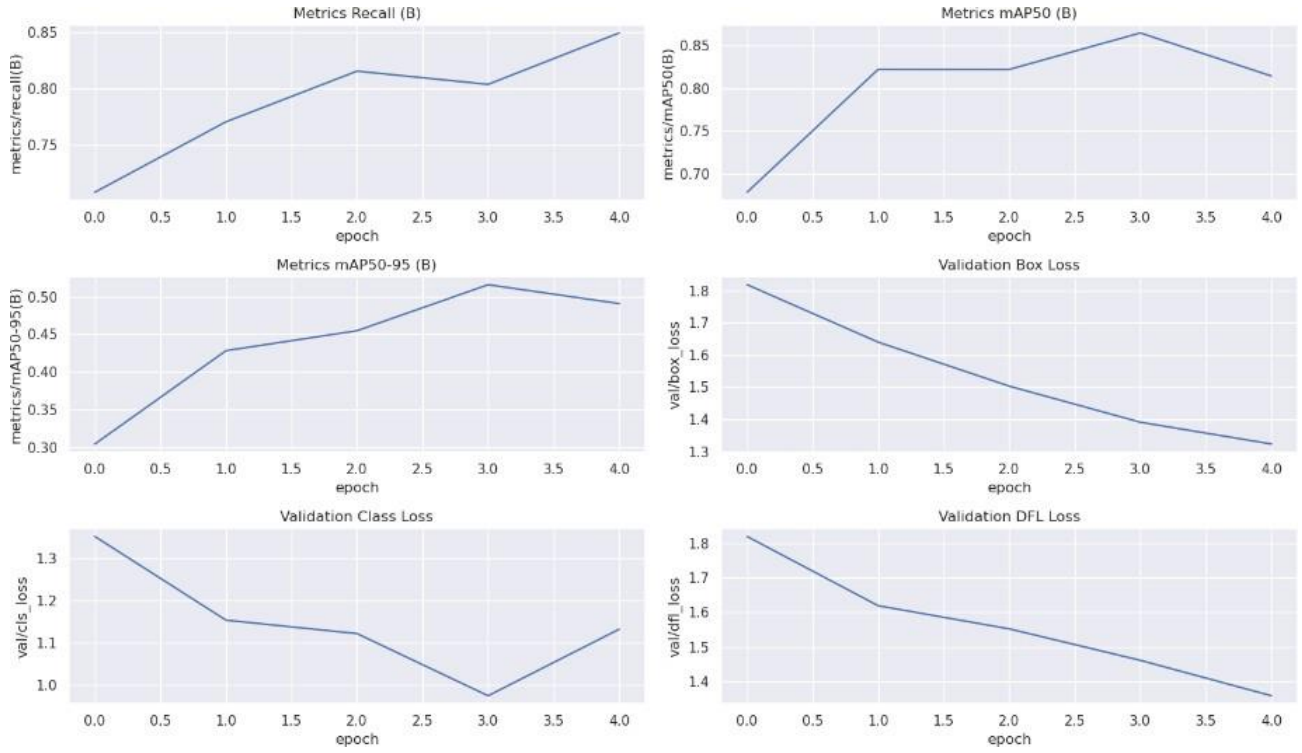


Рис. 8. Метрики при обучении

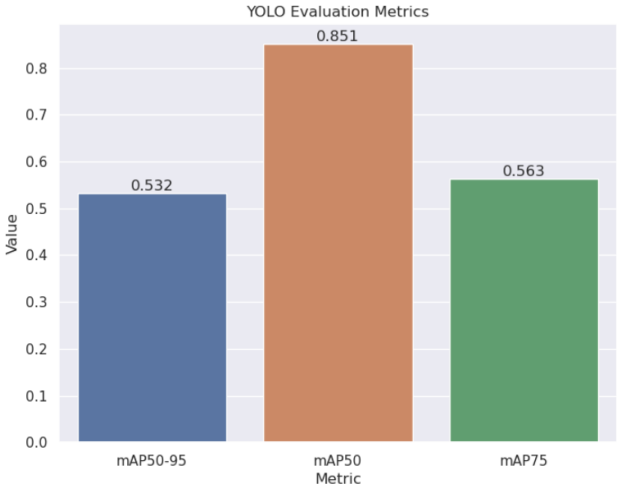


Рис. 9. Метрики лучшей модели

Mean Average Precision (mAP) [17] на рисунке 9 - популярная метрика оценки в задаче обнаружения объектов, включая модель YOLO. Она используется для оценки точности модели обнаружения объектов путем измерения ее способности обнаруживать объекты на изображении, а также точности обнаружения. mAP учитывает как количество правильно обнаруженных объектов, так и качество обнаружения, что делает ее надежной метрикой для оценки производительности моделей обнаружения объектов.

В YOLO mAP особенно важна, так как она измеряет точность модели в обнаружении интересующих объектов. Чем выше значение mAP, тем лучше модель способна идентифицировать объекты на изображении. Поскольку YOLO является моделью обнаружения объектов, разработанной для реального времени, достижение высоких значений mAP критически важно, чтобы модель точно обнаруживала объекты в реальных сценариях. Высокое значение mAP указывает на то, что модель может эффективно идентифицировать объекты и может быть использована с уверенностью в реальных приложениях.

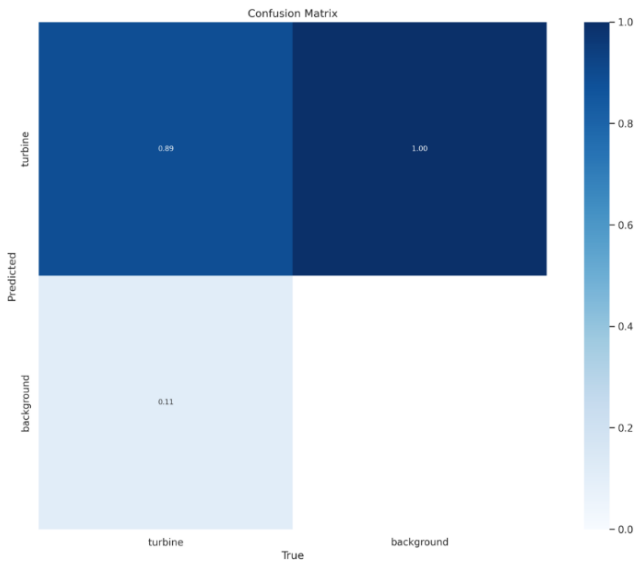


Рис. 10. Матрица ошибок

Однако стоит отметить, что mAP не является идеальной метрикой и имеет свои ограничения. Например, она не учитывает сложность обнаружения определенных типов объектов или важность различных классов объектов.

Тем не менее, она остается широко используемой и ценной метрикой для оценки моделей обнаружения объектов, таких как YOLO. Благодаря ее способности обеспечивать надежную оценку способности модели обнаруживать объекты, mAP является неотъемлемым инструментом как для исследователей, так и для практиков в области компьютерного зрения.

Матрица ошибок [18] (рисунок 10) - полезный инструмент для оценки производительности алгоритмов обнаружения объектов, таких как YOLO. В обнаружении объектов матрица ошибок может быть использована для вычисления различных метрик производительности, таких как точность, полнота и F1-мера. Матрица ошибок — это таблица, которая подводит итоги верных положительных, верных отрицательных, ложных положительных и ложных отрицательных предсказаний, сделанных моделью. В случае обнаружения ветрогенераторов с использованием YOLOv8, матрица ошибок может быть использована для оценки производительности модели в обнаружении ветрогенераторов на изображениях.

Строки матрицы ошибок представляют собой истинные метки (т. е. фактическое наличие или отсутствие ветрогенератора на изображении), а столбцы представляют предсказанные метки (т. е. предсказания модели о наличии или отсутствии ветрогенератора). Истинные положительные (TP) представляют случаи, когда модель правильно предсказывает наличие ветрогенератора, а истинные отрицательные (TN) представляют случаи, когда модель правильно предсказывает отсутствие ветрогенератора. Ложные положительные (FP) представляют случаи, когда модель неправильно предсказывает наличие ветрогенератора, когда его нет, а ложные отрицательные (FN) представляют случаи, когда модель неправильно предсказывает отсутствие ветрогенератора, когда он есть. Исходя из этих значений, мы можем вычислить различные метрики производительности, которые помогут нам оценить производительность модели.

Проведем оценку качества детектирования ветрогенераторов с использованием матрицы ошибок.

$$Recall = \frac{TP}{TP+FN} = 0.89/(0.89 + 0.11) = 0.89$$

$$Precision = \frac{TP}{TP+FP} = 0.89/(0.89 + 1) = 0.47$$

$$F1 = 2\frac{Precision \cdot Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN}$$
$$= 2 * 0.89 / (2 * 0.89 + 0.11 + 0.89) = 0.64$$

Модель продемонстрировала высокие значения точности, что указывает на ее способность точно определять и обнаруживать ветрогенераторы, однако полнота показывает не лучшие результаты. F1-мера лежит между точностью и полнотой и хорошо описывает возможности модели.

V. Заключение

В рамках выполнения работы по обнаружению ветрогенераторов проведено исследование датасета, построения и анализа модели с открытым кодом. Рассмотрены основные наборы данных, на которых обучалась и тестировалась YOLOv8. Был выбран и применен перечень метрик, оценивающих производительность детектирования объектов на изображениях. Эти метрики включали точность, полноту, F1-меру. В ходе работы были изучены и проанализированы нейросетевые методы и алгоритмы машинного обучения, подходящие для этой задачи.

Анализ включал в себя изучение результатов детектирования, определение ошибок, визуализацию предсказаний и общую эффективность модели. Это позволило оценить, насколько успешно модель обнаруживает ветрогенераторы.

Обобщая результаты, можно заключить, что использование модели YOLOv8 для обнаружения ветрогенераторов демонстрирует перспективность и эффективность в данной задаче, превышая показатели ручной оценки.

Литература

[1] Pierce, S. G., Burnham, K. C., Zhang, D., McDonald, L., MacLeod, C. N., Dobie, G., Summan, R. McMahon D.“Quantitative inspection of wind turbine blades using UAV deployed photogrammetry”

[2] Sadekov, R.N. et al. (2017) ‘Road sign detection and recognition inpanoramic images to generate navigational maps’, 2017 24th SaintPetersburg International Conference on Integrated NavigationSystems (ICINS) [Preprint]. doi:10.23919/icins.2017.7995611

[3] Chisulo Mukabe, Nalina Suresh, Valerians Hashiyana, Titus Haiduwa, William Sverdlik. “Object Detection and Classification Using Machine Learning Techniques: A Comparison of Haar Cascades and Neural Networks” (August 2021)

[4] N. S. Guzhva, V. E. Prun, V. V. Postnikov, M. G. Lobanov, R. N.Sadekov and D. L. Sholomov, "Using 3D Object Detection DNN in anAutonomous Tram to Predict the Behaviour of Vehicles in the RoadScene," 2022 29th Saint Petersburg International Conference onIntegrated Navigation Systems (ICINS), Saint Petersburg, RussianFederation, 2022, pp. 1-6, doi: 10.23919/ICINS51784.2022.9815388

[5] Zhong-Qiu Zhao, Member, IEEE, Peng Zheng, Shou-tao Xu, and Xindong Wu, Fellow, IEEE. “Object Detection with Deep Learning: A Review” (16 Apr 2019)

[6] Zhengxia Zou, Keyan Chen, Zhenwei Shi, Member, IEEE, Yuhong Guo, and Jieping Ye, Fellow, IEEE. “Object Detection in 20 Years: A Survey” (18 Jan 2023)

[7] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi. “You Only Look Once: Unified, Real-Time Object Detection” (9 May 2016)

[8] Suorong Yang, Weikang Xiao, Mengchen Zhang, Suhan Guo, Jian Zhao, Furao Shen. “Image Data Augmentation for Deep Learning: A Survey” (5 Nov 2023)

[9] Keiron O’Shea and Ryan Nash. “An Introduction to Convolutional Neural Networks” (2 Dec 2015)

[10] Zijie J. Wang, Robert Turko, Omar Shaikh, Haekyu Park, Nilaksh Das, Fred Hohman, Minsuk Kahng, and Duen Horng (Polo) Chau. “CNN EXPLAINER: Learning Convolutional Neural Networks with Interactive Visualization” (28 Aug 2020)

[11] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich. “Going deeper with convolutions” (17 Sep 2014)

[12] Chien-Yao Wang, Hong-Yuan Mark Liao, I-Hau Yeh, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh. “CSPNET: A NEW BACKBONE THAT CAN ENHANCE LEARNING CAPABILITY OF CNN” (27 Nov 2019)

[13] Jianbiao Mei, Yu Yang, Mengmeng Wang, Xiaojun Hou, Laijian Li and Yong Liu. “PANet: LiDAR Panoptic Segmentation with Sparse Instance Proposal and Aggregation”

[14] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. “Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition” (18 Jun 2014)

[15] T. Lin, M. Maire, S. J. Belongie, Hays et. al. “Microsoft COCO: Common Objects in Context. European Conference on Computer Vision” (2014)

[16] Zhora Gevorgyan. “SIoU Loss: More Powerful Learning for Bounding Box Regression” (25 May 2022)

[17] Paul Henderson, Vittorio Ferrari. “End-to-end training of object class detectors for mean average precision” (12 Jul 2016)

[18] Richard Evans. “Confusion Matrices and Accuracy Statistics for Binary Classifiers Using Unlabeled Data: The Diagnostic Test Approach” (26 Aug 2022)

Сведения об авторах

Хонер Павел Дмитриевич - компьютерное зрение, машинное обучение, нейронные сети, python (numpy, pandas, pytorch), базовое владение - C++
email: Khonerworki@gmail.com

Антонов Илья Андреевич - классическое машинное обучение и нейронные сети применительно к задачам табличной классификации (в частности, обнаружение вторжений в сетях SDN и IoT), python, tensorflow,
email: antonov.ia240701@yandex.ru

Селезнев Иван Андреевич - искусственный интеллект в геймдизайне/распозновании лица и микромимики. Базовые знания python, c++, c#, vba, навыки граф. Дизайнера
email: m1902948@edu.misis.ru

Исаченко Михаил Константинович - машинное обучение и нейронные сети применительно к вопросам кибербезопасности (детекция фишинга, обнаружения вторжений в сетях), C++, Qt, Python, Swift, Keras
email: isachenko.mikhail.k@gmail.com

Кожухов Александр Алексеевич - компьютерное зрение, глубокое обучение с подкреплением, робототехника. Python, PyTorch, Базовое знание C++, CUDA.
email: kozhukovv@yandex.ru

Бугаков Никита Игоревич - искусственный интеллект, как инструмент облегчения разработки игр и ПО (в частности, создание ассетов, расширение возможностей НПС, применение в бухгалтерии) ,C# , unity , vba , PIX RPA , UiPath.
email: m1900660@edu.misis.ru

Лойко Антон Геннадьевич - искусственный интеллект, мобильная разработка и бизнес аналитика. C++, Kotlin, Python.
email: loikoanton@yandex.ru

Измайлов Лев Сергеевич - классическое машинное обучение, нейронные сети для задач классификации, кластеризации, обработки текста (разбор трансформеров) и работа с данными. Python, pytorch.
email: izmaylovle0@gmail.com

Ступина Анастасия Александровна - машинное обучение и нейронные сети, компьютерное зрение, Python, PyTorch, Tensorflow, SQL, базовое знание C++.
email: stupinaaa99@gmail.com

Береснев Денис Викторович - ai шахматный движок, компьютерное зрение, обработка естественного языка, фуллстек разработка. Js, python, c++.
email: denisberesnev59@gmail.com

Личко Дмитрий Алексеевич - машинное обучения для трейдинга/облегчения разработки ПО/анализа данных, веб-разработка, JS, Python, Ruby, C++.
email: lichko2002@mail.ru

Вершинин Кирилл Александрович – искусственный интеллект, web-разработка, системная и бизнес-аналитика. Python, JS, C++.

email: evenmares@gmail.com

Фомина Анна Александровна - машинное обучение, нейронные сети, компьютерное зрение, анализ данных, Python, PyTorch, Tensorflow, SQL. Базовое знание C++.
email: annafomina2555@gmail.com

Леонов Иван Юрьевич - классическое машинное обучение, анализ временных рядов, web-разработка. Python, JS.
email: vanleo528@yandex.ru

Коновалов Матвей Алексеевич - анализ данных, машинное обучение, нейронные сети, NLP и CV, python (numpy, pandas, seaborn, Tensorflow, Pytorch, ultralytics, os), git, SQL.
email: matvei.konovalov@mail.ru

Антипов Иван Илич - анализ данных, машинное обучение, нейронные сети. Владение Python, базовое знание C++, C#, Java, SQL. Работа с технологиями greenplum, patroni, apache hadoop, apache airflow, clickhouse, PostgreSQL.
email: somaksa@mail.ru

Лоткова Дарья Васильевна – нейронные сети, искусственный интеллект, как инструмент современного искусства, UX/UI дизайн, web-разработка, UX Researcher. Базовые знания python, C++, C#, SQL.
email: dv.lotkova@yandex.ru

Карякин Алексей Владимирович - машинное обучение, нейронные сети, аналитика, backend-разработка. Python, PyTorch, Java, C++.
email: al.kariackin2017@yandex.ru

Хуако Виктор Олегович - нейронные сети, обработка естественного языка, компьютерное зрение, python, backend-разработка , базовое знание c++ и cuda,
email: m2305334@edu.misis.ru

Терзиян Грант Игоревич - машинное обучение, нейронные сети, NLP. Python, C/C++, SQL. email: grant.terz@gmail.com

Корчевский Александр Сергеевич - ИИ, DevOps, C++, Python.
email: just4n4cc@yandex.ru

Подгорный Данила Александрович - компьютерная графика, нейронные сети, геймдев. C++, OpenCL, Unreal Engine, Python, SQL, Docker.
email: d.podgornyy@inbox.ru

Абакумов Александр Антонович - нейронные сети, компьютерная графика, базовое знание C++ и python, frontend-разработка
email: solitude11111@mail.ru

Ерещенко Алексей Геннадьевич - компьютерное зрение, python, SQL, базовое владение C#, docker,
email: aleksey-000@mail.ru

Злакоманов Павел Евгеньевич - алгоритмы, backend, C#, PHP, python, нейронные сети,

email: pavel.zlakomanov@mail.ru
Алексеев Игорь Борисович - компьютерное зрение, Python, C++, SQL, Backend-разработчик,
email: igolexeev@mail.ru

Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях, 2023: Сборник статей научно-технического семинара студентов. Вып. 1 / Под ред. А.Р. Ефимова— М.: НИТУ «МИСИС», 2023.— 168 с.: табл., ил., цв. ил.

Редакционная коллегия: Ефимов А.Р., Бакулев К.С., Садеков Р.Н., Мишуров С.С.
Редактор: Садеков Р.Н.
Компьютерная верстка: Садеков Р.Н.