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-ТЕХНОЛОГИЧЕСКОГО СЕМИНАРА КАФЕДРЫ «ИНЖЕНЕРНОЙ КИБЕРНЕТИКИ» НА ТЕМУ «ИСКУСТВЕННЫЙ ИНТЕЛЛЕКТ В ПРОМЫШЛЕННЫХ, КОММЕРЧЕСКИХ, МЕДИЦИНСКИХ И ФИНАНСОВЫХ ПРИЛОЖЕНИЯХ»

Москва, 2025

УДК 0004.8
ББК 32.813.5

Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях, 2025: Сборник статей научно-технического семинара. Вып. 4 / Под ред. А.Р. Ефимова - М.: НИТУ «МИСИС», 2025. - 113 с.: табл., ил., цв.ил.

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

Дата проведения семинара 30 мая 2025 г.
Редакционная коллегия: Ефимов А.Р., Бакулев К.С., Садеков Р.Н., Мишуров С.С.
Редактор: Садеков Р.Н.
Компьютерная верстка: Садеков Р.Н.

Рецензенты: Садеков Р.Н. д.т.н., доцент, профессор кафедры инженерной кибернетики «МИСИС», Тарханов И.А. к.т.н., доцент кафедры инженерной кибернетики НИТУ «МИСИС», Курочкин И.И. к.т.н., доцент кафедры инженерной кибернетики НИТУ «МИСИС»

Содержание

ГРНТИ 20.53.21, 20.53.23, 20.53.25
ASJC 1707, 1712, 1706
OECD 01.02.EP

Распознование самокатов в реальном времени с помощью YOLO: сравнительный анализ YOLOv10, YOLOv11

Л.Е. Алексеев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2413819@edu.misis.ru

В условиях стремительного роста числа электросамокатов в городских зонах увеличивается рисковая нагрузка на пешеходов и другие категории участников движения. По Правилам дорожного движения РФ — «переезжать по пешеходному переходу на электросамокате без спешивания запрещается». Мы формализуем задачу автоматической фиксации именно этого нарушения: «самокатчик едет по «зебре» с человеком на самокате». Сопоставляются две последние версии высокоэффективных детекторов семейства YOLO — YOLOv9 и YOLOv11 — по метрикам точности (mAP@0.5, mAP@0.5–0.95), вычислительным затратам (параметры, GFLOPs). Экспериментальные результаты демонстрируют, что YOLOv11 обеспечивает более высокую точность и скорость, тогда как YOLOv9 показывает конкурентоспособную эффективность на edge-устройствах.

Ключевые слова — Компьютерное зрение, Детекция электросамокатов, Нарушения ПДД, YOLOv9, YOLOv11, mAP, Инференс в реальном времени, Пешеходный переход, Индивидуальные транспортные средства, Нарушение ПДД с использованием ИТС

Рост популярности средств индивидуальной мобильности (СИМ), к которым относятся электросамокаты, заставил законодателя актуализировать ПДД РФ. По пункту 24.2 раздела 24 ПДД РФ «движение на СИМ разрешается по тротуару, пешеходной дорожке при отсутствии велосипедных дорожек… и при обязательном спешивании перед пересечением проезжей части». Однако на практике нередко фиксируются случаи, когда самокатчик переезжает пешеходный переход, оставшись «сидеть» на самокате, что создаёт опасность для всех участников движения.

Автоматизация контролируемого видеонаблюдения требует надёжных алгоритмов детекции таких эпизодов в реальном времени. Семейство YOLO (You Only Look Once) заслужило признание за единичный проход по изображению, объединяющий локализацию и классификацию объектов. В данной работе мы рассматриваем две свежайшие версии — YOLOv9 и YOLOv11 — обосновываем их выбор для нашей задачи и проводим детальный сравнительный анализ.

I. Нормативно-правовая база

Средства индивидуальной мобильности (СИМ) — согласно ПДД РФ, это транспортные средства на колёсах или роликах с двигателем, не превышающие 35 кг, включая электросамокаты.

Права и обязанности — пользователи СИМ приравнены к пешеходам, при этом перед пересечением проезжей части по пешеходному переходу обязаны спешиваться и вести устройство рядом.

Ответственность — нарушение рассматривается как пешеходное нарушение ПДД, но при наличии автоматической фиксации может приводить к штрафам и блокировке учётной записи в сервисах проката.

Таким образом, задача детекции «самокат превысил требования ПДД» формализуется как одновременное появление на кадре:

- самоката, целиком заходящего в границы «зебры»;
- объекта «человек» расположенного на раме/седле самоката;
- отсутствие спешивания (положение тела над плечами, IoU человека с самокатом выше порога).

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей, использовались один личный и один открытый набор данных

A. Публичный набор данных (ScooterDet)

Набор данных ScooterDet содержит 2013 изображений с 11 классами дорожных объектов: "человек", "автомобиль", "грузовик", "автобус", "светофор", "пожарный гидрант", "знак стоп", "скамейка" и "самокат". Всего в наборе 11 011 аннотаций ограничивающих рамок. Данные собраны с помощью очков Tobii Pro Glasses 2, установленных на самокате Segway NineBot, при движении от кампуса Университета Вирджинии до городской зоны Шарлотсвилля, штат Вирджиния, США. Изображения извлечены из видеозаписей, снятых этими очками, и очищены от низкокачественных кадров и изображений без релевантных объектов.

Изначально набор данных содержал аннотации в формате JSON, созданные с помощью инструмента разметки LabelMe. Однако для обеспечения совместимости с фреймворком обучения YOLO потребовалась конвертация аннотаций в стандартный формат YOLO. Этот процесс включал преобразование координат ограничивающих рамок и классов объектов в числовые значения, соответствующие требованиям формата YOLO.

Следовательно, мы преобразовали датасет таким образом, чтобы он был представлен в стандартной разметке YOLO:

- Класс объекта (номер класса).

- Координаты центра ограничивающей рамки (x, y) в нормализованных значениях.
- Ширина и высота рамки относительно размеров изображения.

После преобразования данные были случайным образом разделены на три подмножества:

- Обучающая выборка – 60% (1207 изображений)
- Валидационная выборка – 20% (402 изображения)
- Тестовая выборка – 20% (404 изображения)

На следующем этапе данные были дополнительно обработаны (аугментация), чтобы повысить их разнообразие и устойчивость к реальным условиям, включая изменения освещения, ракурса и фона.

Модели YOLO проходили обучение с использованием заранее натренированных весов (transfer learning), полученных на наборе данных COCO. Изображения были приведены к разрешению 640 × 640 пикселей, соответствующему входным требованиям YOLO.

Каждое изображение сопровождается текстовым файлом аннотаций, где указаны:

- Класс объекта (номер класса).
- Координаты центра ограничивающей рамки (x, y) в нормализованных значениях.
- Ширина и высота рамки относительно размеров изображения.

Рис. 1. Примеры изображений ScooterDet

B. Набор CDNet

Предназначение и сбор. CDNet (Crosswalk Detection Network) — специализированный датасет для детекции пешеходных переходов, предложенный Zhang et al. (2022) для встраиваемых решений на Jetson Nano . Содержит ≈1500 уникальных изображений городских улиц с разметкой «зебры» (crosswalk) в различных погодных и световых условиях.

Типы аннотаций. В CDNet представлены два вида аннотаций:

1. Bounding Box (Axis-Aligned BB) для грубой локализации всего перехода.
2. Oriented Bounding Box (OBB), задающий четырёхточечный полигон, точно описывающий форму «зебры», что важно при съёмке под углом.

Для унификации с остальными данными эти аннотации были переконвертированы в формат YOLO, при этом:

- каждому OBB-боксу вычислялся минимальный axis-aligned bbox;
- классы сведены к двум: 12 — crosswalk, 13 — guide_arrows (указатели направления);
- координаты нормализованы по формуле YOLO.

Подготовка выборок. Датасет разбит на тренировочную (70 %, ≈1050 изобр.), валидационную (15 %, ≈225 изобр.) и тестовую (15 %, ≈225 изобр.) части. Дополнительно применялась фотометрическая аугментация:

- имитация бликов и теней;
- шум Гаусса и JPEG-артефактов;
- небольшие геометрические искажения (перспектива ±5 %).

Окончательные кадры также приведены к размеру 640×640, обеспечивая согласованность со ScooterDet.

Рис 2. Примеры изображений «Зебры» из датасет CDNet

C. Доработка собственной обучающей выборки

Сбор и фильтрация. Для моделирования реального городского трафика была отобрана съёмка с бортовых камер автомобилей Tesla, курсировавших по улицам Москвы. Общий объём видеозаписей составил ~128 ГБ (разрешение 1920×1080, 30 FPS).

Необходимо было получить кадры и отфильтровать их содержимое: на вход поступали 4 видео с разных ракурсов, необходимо было подобрать нужные видео, найти объекты, оценить степень освещенности и наличие объектов для детекции, иногда видео были пустые в связи с тем, что автомобиль записывает и регулярные поездки. В связи с тем, что видео записывается 30 кадров в секунду, было сокращено с помощью CVAT до 5% кадров от одного видео, что существенно облегчило процесс разметки и обработки.

Разметка проводилась в CVAT по следующей схеме:

- класс 11 — scooter (как стоящий, так и движущийся);
- класс 0 — person;
- класс 12 — crosswalk;
- класс 13 — guide_arrows.

Каждая bounding box записывалась в формате YOLO с нормализованными координатами. Среднее время разметки одного кадра — 45 с.

Формирование выборок. После завершения разметки получены:

- 4000 кадров с аннотациями, из которых:
  - тренировочная выборка — 4000 изображений;
  - валидационная выборка — 400 изображений;

При разделении учитывалось равномерное распределение по трём основным классам (scooter, person, crosswalk), а также по локациям (центральные улицы, окраины, парковки), что гарантирует устойчивость моделей к смене фона и плотности пешеходов.

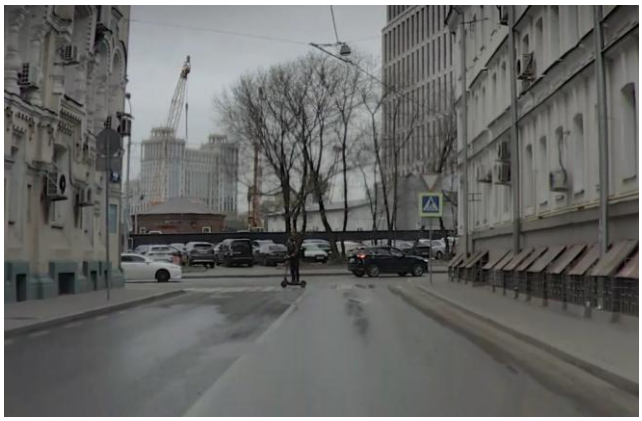


Рис. 3. Примеры изображения с нарушением ПДД

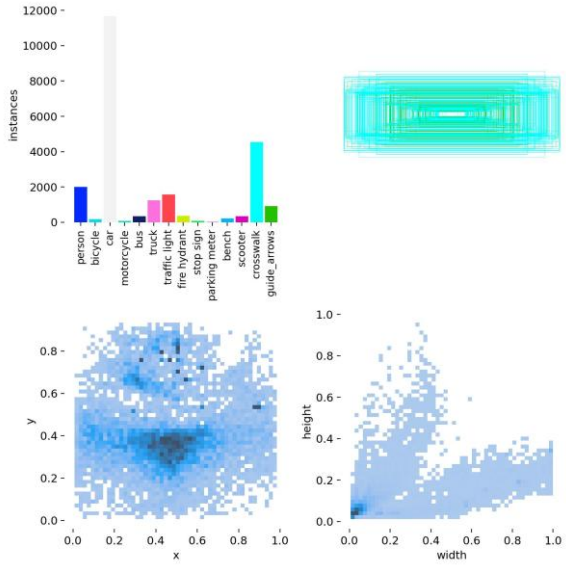


Рис 4. Распределение объеденных выборок

III. Архитектурные особенности нейросетей YOLO

A. *YOLOv9*

YOLOv9 — релиз который состоялся в феврале 2024. Это архитектурное решение добавляет Programmable Gradient Information (PGI) — сохраняет и передаёт ключевые градиентные сигналы между слоями, улучшая глубокое обучение в лёгких сетях. А также появляется Generalized Efficient Layer Aggregation Network (GELAN) — эффективная агрегация признаков разной глубины, сокращающая информационные потери и SPPF-блок — мульти-масштабный пуллинг для мелких/крупных объектов. Также существует еще одно преимущество Dual Head — независимые ветви для разных размеров объектов, которые помогают решать задачи сегментации разных объектов.

Таким образом, YOLOv9 занимает нишу «легковесных высокоточных» решений, оптимизированных под edge-устройства с ограниченными ресурсами. Выбор этой модели основан на адаптации к edge-устройствам, что позволяет запускать модели даже на камерах видео наблюдений.

B. *YOLOv11*

YOLOv11 — свежайшая версия октябрь 2024 компании Ultralytics, в данной архитектуре они добавили:

1. C3k2-блоки (Cross Stage Partial, ядро 2×2) — ускоряют вычисления при сохранении качества признаков.
2. Parallel Spatial Attention (C2PSA) — два модуля PSA, улучшающих фокусировку на ключевых регионах (важно для частично скрытых объектов).
3. Оптимизированная голова (Conv-BN-SiLU) — повышает стабильность обучения.
4. OBB-детекция и Instance Segmentation — расширенные возможности для ориентированных и сегментных задач.

Так, YOLOv11 балансирует «высокую точность» и «низкую задержку», что в свою очередь позволяет уверенно детектировать даже сложные случаи перекрытия самокатчика и «зебры», например другими пешеходами.

IV. Обучение

Для переобучения обеих моделей использовалась стандартная рабочая станция с видеокартой NVIDIA GeForce RTX 3070 (8 ГБ GDDR6) — потребительский ускоритель уровня «домашнего ПК».

- YOLOv11 обучалась в течение 60 эпох, что заняло ≈26 часов.
- YOLOv9, обладая более лёгкой архитектурой, прошла 60 эпох за ≈18 часов на той же конфигурации.

V. Сравнение

A. *Результаты*

В аспекте точности YOLOv11s (mAP@0.50 = 0.63) превосходит YOLOv9s (mAP@0.50 = 0.60) на +5 % благодаря C2PSA и C3k2, улучшающим выделение мелких деталей «зебры» и человека на самокате. И по mAP@0.50–0.95: 0.25 против 0.22, что составляет прирост +13.6 %. По скорости инференса YOLOv11s обрабатывает одно изображение за 1.4 мс, тогда как YOLOv9s требуется 2.1 мс (почти на 33 % дольше) — важный фактор для систем реального времени на embedded. По вычислительной эффективности, несмотря на большее число параметров у YOLOv11s, у неё ниже GFLOPs, что указывает на более оптимизированную архитектуру.

Очевидно, что более новая версия модели YOLO практически во всем превосходит своего предшественника за исключением скорости обучения, в промышленных масштабах и при должной вычислительной мощности этот недостаток может быть устранен.

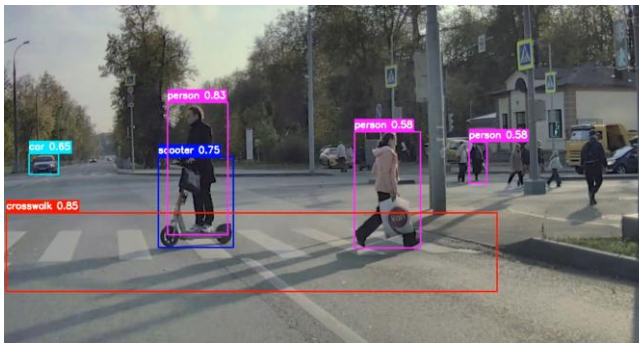


Рис 5. Демонстрация работы YOLOv11

VI. Заключение

В рамках данной работы был существенно дополнен и расширен корпус данных для задачи автоматического обнаружения нарушений ПДД электросамокатчиками: помимо публичного ScooterDet и специализированного CDNet для crosswalk, мы объединили собственную выборку из московских камер Tesla, доведя общую выборку до почти 10 000 изображений с равномерным распределением по ключевым классам («scooter», «person», «crosswalk», «guide_arrows»). Такое расширение обеспечило универсальность датасета, позволив не только надёжно детектировать езду «на самокате по зебре», но и заложить основу для обучения моделей на обнаружение других нарушений — движения вдвоём, выезда на тротуар, игнорирования дорожных знаков и т. п.

Сравнительный анализ двух последних версий семейства YOLO показал, что YOLOv11 обеспечивает наилучший баланс между точностью (mAP@0.5 +4.7 % и mAP@0.5–0.95 +4.2 % относительно YOLOv9), скоростью инференса (до 33 % выигрыша) и вычислительной эффективностью (ниже GFLOPs при росте числа параметров). При этом YOLOv9 остаётся привлекательным вариантом для задач со жёсткими ресурсными ограничениями, демонстрируя достойную точность и заметно меньшую нагрузку на железо.

В целом, сочетание универсального объединённого датасета и современных версий YOLO позволяет создавать гибкие системы видеоконтроля, способные быстро адаптироваться к новым видам нарушений ПДД.

Литература

[1] Правила дорожного движения РФ (утв. постановлением Правительства РФ от 23 октября 1993 г. № 1090, в ред. от 6 октября 2022 г. № 1769) «Дополнительные требования к движению велосипедистов и средств индивидуальной мобильности» // Consultant.ru. Available at: https://www.consultant.ru/document/cons_doc_LAW_2709/b11240808fefa9a32c03b5b7c81829af379a790e/ (Accessed: 12 March 2025)

[2] Sher.media (2023) «Теперь все пользователи средств индивидуальной мобильности должны спешиваться перед тем, как перейти дорогу» // Sher.media. Available at: https://sher.media/teper-vse-polzovateli-sredstv-individualnoj-mobilnosti-dolzhny-speshivatsya-pered-tem-kak-perejti-dorogu/ (Accessed: 28 March 2025)

[3] Chen D., Hosseini A., Smith A., Nikkhah A. F., Heydarian A., Shoghli O., Campbell B. (2024) “Performance Evaluation of Real-Time Object Detection for Electric Scooters” // arXiv. Available at: https://arxiv.org/abs/2405.03039 (Accessed: 10 April 2025)

[4] Dwivedi N. (2024) “YOLOv11: The Next Leap in Real-Time Object Detection” // Analytics Vidhya Blog. Available at: https://www.analyticsvidhya.com/blog/2024/10/yolov11/ (Accessed: 5 May 2025)

[5] Ultralytics (2024) “YOLOv9 & YOLOv11 Models Documentation” // Ultralytics Docs. Available at: https://docs.ultralytics.com/models/ (Accessed: 19 April 2025)

[6] Rao N. (2024) “YOLO Explained: From v1 to v11” // Viso.ai. Available at: https://viso.ai/computer-vision/yolo-explained/ (Accessed: 22 April 2025)

[7] Jocher G., Stoken A., Changyu Lin, NanoCode Team (2023) “ultralytics/YOLOv5: Implementation of YOLOv5 in PyTorch” // GitHub. Available at: https://github.com/ultralytics/yolov5 (Accessed: 15 March 2025)

[8] Wang C.-Y., Bochkovskiy A., Liao H.-Y M. (2023) “YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors” // arXiv. Available at: https://arxiv.org/abs/2207.02696 (Accessed: 20 March 2025)

[9] Jocher G., Wang C., Du N. (2022) “YOLOv8: A Trainable Ensemble of YOLO Designs” // Ultralytics Docs. Available at: https://github.com/ultralytics/ultralytics (Accessed: 25 April 2025)

[10] Lin T.-Y., Maire M., Belongie S., Hays J., Perona P., Ramanan D., Dollár P., Zitnick C. L. (2015) “Microsoft COCO: Common Objects in Context” // Computer Vision – ECCV 2014. Available at: https://cocodataset.org (Accessed: 5 April 2025)

[11] Everingham M., Gool L. V., Williams C. K. I., Winn J., Zisserman A. (2010) “The PASCAL Visual Object Classes (VOC) Challenge” // International Journal of Computer Vision. Available at: http://host.robots.ox.ac.uk/pascal/VOC/ (Accessed: 10 April 2025)

[12] Cordts M., Omran M., Ramos S., Rehfeld T., Enzweiler M., Benenson R., Franke U., Roth S., Schiele B. (2016) “The Cityscapes Dataset for Semantic Urban Scene Understanding” // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Available at: https://www.cityscapes-dataset.com (Accessed: 20 March 2025)

[13] AI2024 (n.d.) “TorchVision Object Detection Models” // PyTorch Documentation. Available at: https://pytorch.org/vision/stable/models.html#object-detection (Accessed: 1 May 2025)

ГРНТИ 28.23.15, 28.23.37
ASJC 1707, 1702, 2308
OECD 01.02.EP

Глубокие нейросетевые подходы к сегментации нефтяных разливов

С. Д. Киселев
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m1900033@edu.misis.ru

А. В. Алтунян
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m1801726@edu.misis.ru

***Аннотация* — сегментация объектов на спутниковых и аэрофотоснимках используется в задачах экологического мониторинга для выделения зон загрязнения. В настоящем исследовании рассматриваются два подхода к задаче сегментации нефтяных разливов: сверточная нейросеть Mask R-CNN, а также архитектура Swin V2 Tiny, использованная как компонент трансформерной модели сегментации на основе извлечения признаков и оконного внимания [1]. В качестве исходных данных применялись изображения, маски объектов и таблица с цветовой разметкой классов. Основное внимание уделено оценке качества сегментации по метрикам Precision, Recall, F1-score и IoU, а также визуальному анализу предсказаний. Методы были протестированы на наборе данных с изображениями реальных загрязнений, что позволило изучить особенности выделения разливов в различных условиях освещенности, формы и контраста с фоном. Потенциальные направления дальнейшего применения могут быть связаны с автоматизацией мониторинга водных акваторий и снижением времени реагирования на инциденты.**

Ключевые слова — Компьютерное зрение, Сегментация изображений, Нефтяные разливы, Трансформеры, Swin V2, Mask R-CNN, Экологический мониторинг, Распознавание образов

I. Введение

Распознавание и анализ объектов на изображениях в реальном времени является одной из ключевых задач в области искусственного интеллекта и компьютерного зрения. Особый интерес представляет применение этих технологий в сфере экологического мониторинга, в частности для автоматического выявления нефтяных разливов на водной поверхности [2], [3]. Подобные инциденты представляют собой серьезную угрозу для морских экосистем, что требует оперативного и точного обнаружения зон загрязнения [4].

Современные методы глубокого обучения, в частности сверточные нейронные сети (CNN), обеспечивают высокую точность в задачах классификации и сегментации изображений [5]. Одной из наиболее эффективных архитектур для задач сегментации является Mask R-CNN, которая позволяет не только определять наличие объекта на изображении, но и выделять его точные границы.

В настоящей работе применяется модель *maskrcnn_resnet50_fpn* — одна из реализаций Mask R-CNN, сочетающая в себе мощность глубоких сверточных слоев и пирамидальную архитектуру признаков (Feature Pyramid Network), что позволяет эффективно обрабатывать объекты различных размеров. Обучение модели проводилось на наборе данных, включающем изображения нефтяных пятен и соответствующие маски, что обеспечивает возможность точной сегментации зон разлива.

Дополнительно в качестве самостоятельной модели сегментации была обучена трансформер-архитектура Swin V2 Tiny (*swinv2-tiny-patch4-window8-256*). Модель выбрана засчет высокой эффективности на высокоразрешенных данных и способности учитывать локальные и глобальные контексты через оконный механизм внимания.

Целью данной работы является исследование эффективности моделей Mask R-CNN и Swin V2 Tiny в задаче сегментации нефтяных разливов, а также анализ полученных результатов с точки зрения точности, полноты и визуальной интерпретируемости. Работа направлена на демонстрацию того, как технологии искусственного интеллекта могут быть применены для решения актуальных задач в области охраны окружающей среды.

II. Наборы данных

Для обучения и оценки моделей, рассматриваемых в данной работе, использовался специализированный датасет, предоставленный коллегой-экспертом по компьютерному зрению. Коллекция содержит 1 040 полноразмеченных RGB-снимков, охватывающих три релевантных класса: «нефть», «вода» и «прочее» (фоновые или нерелевантные объекты) [6]. Аннотация выполнена вручную в формате COCO — на каждом кадре полигональными масками точно выделены участки разливов нефти и водной поверхности, что обеспечивает пиксельную точность сегментации [7].

Для корректного контроля переобучения выборка была разбита на три непересекающихся подмножества в пропорции 65% / 15% / 20%. Обучающая часть (Train) включает 672 изображения и используется для оптимизации весов нейронной сети. Валидационный набор (Val) применяется для подбора гиперпараметров и ранней остановки. Итоговая проверка обобщающей способности проводится на независимом тестовом комплекте (Test), содержащем 208 изображений, которые модель не видела на предыдущих этапах. Подобная стратифицированная схема обеспечивает объективное сравнение с существующими решениями и воспроизводимость полученных результатов.

III. Подготовка данных

Для корректного обучения нейросетевой модели необходимо обеспечить структурированную подачу входных данных и целевых меток, а также повысить устойчивость модели к разнообразию визуальных условий. На данном этапе были реализованы процедуры подготовки датасета и расширения обучающей выборки засчет аугментаций.

i. Подготовка датасета и генерация целевых структур

Для обучения модели использовался пользовательский датасет, включающий RGB-изображения и соответствующие им цветные маски. Маски содержали разметку по трем категориям: нефтяной разлив (цвет [255, 0, 124]), водная поверхность ([51, 221, 255]) и прочие элементы ([255, 204, 51]).

Рис. 1. Пример исходного снимка и соответствующей маски-аннотации. Голубым цветом обозначена водная поверхность, розовым – нефтяные разливы, желтым – прочие объекты

В процессе чтения изображений и масок выполнялось преобразование маски в многоканальный numpy-массив и бинаризация каждого класса по заданному цветовому коду. Для каждого бинарного слоя маски с помощью метода cv2.findContours извлекались контуры объектов, после чего рассчитывались прямоугольные ограничивающие рамки (bounding boxes). Для каждого найденного объекта формировались:

- метка класса (label) — целочисленный идентификатор;
- рамка (bounding box) — прямоугольная область, минимально охватывающая контур;
- маска (mask) — бинарное изображение, выделяющее объект пиксельно.

Все данные агрегировались в словарь, содержащий тензоры boxes, labels и masks, необходимый для обучения Mask R-CNN.

ii. Аугментации и трансформации

С целью повышения обобщающей способности модели к различным условиям съемки (освещение, форма разлива, контрастность), использовались пространственные и фотометрические аугментации. Для их реализации применялась библиотека Albumentations, предоставляющая высокоэффективные функции с поддержкой PyTorch [8], [9].

В частности, были применены следующие преобразования:

- случайное горизонтальное отражение (flip), имитирующее изменение направления обзора;
- изменение яркости и контрастности, компенсирующее влияние погодных условий и времени суток;
- масштабирование и обрезка, усиливающее устойчивость модели к различию в разрешении и зуме;
- нормализация входного изображения согласно стандартным статистикам ImageNet.

Аугментации синхронно применялись к изображениям и маскам.

IV. Нейросетевые архитектуры

В рамках настоящей работы для задачи сегментации нефтяных разливов на водной поверхности использована сверточная нейронная сеть Mask R-CNN с базовой архитектурой *maskrcnn_resnet50_fpn*, реализованной в библиотеке torchvision. Данная модель была выбрана засчет своей высокой точности в задачах instance-сегментации и способности выявлять объекты различных форм и размеров на фоне сложной структуры, а также засчет широкого применения в научной и прикладной практике, включая задачи экологического мониторинга.

Дополнительно использована трансформер-архитектура Swin V2 Tiny (*swinv2-tiny-patch4-window8-256*). Эта модель выбрана благодаря сочетанию компактности и высокой точности: оконный механизм внимания эффективно объединяет локальные детали и глобальный контекст, что особенно важно при анализе разливов с выраженной текстурной разнородностью.

A. Mask R-CNN

Архитектура Mask R-CNN расширяет модель Faster R-CNN засчет добавления дополнительной ветви для предсказания масок объектов на уровне пикселей. Она

сохраняет двухступенчатую структуру, характерную для моделей семейства R-CNN:

1. Первый этап — определение потенциальных регионов, содержащих объекты (Region Proposal Network, RPN);
2. Второй этап — классификация найденных регионов, уточнение ограничивающих рамок и генерация бинарных масок объектов.

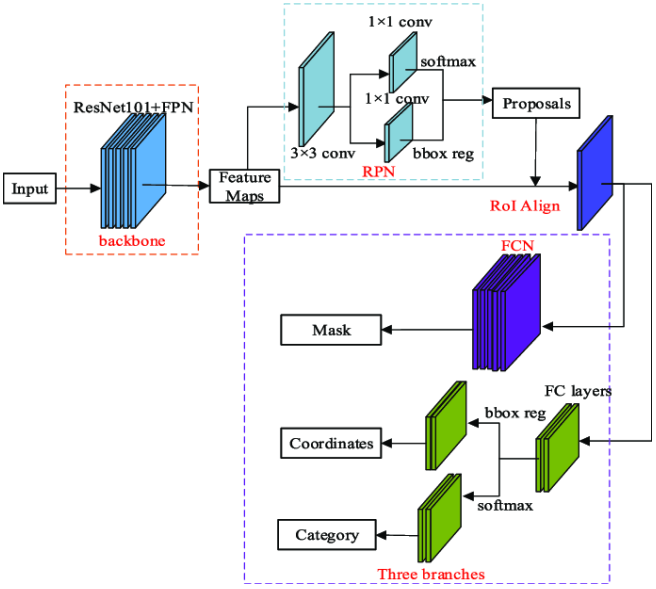


Рис. 2. Архитектура Mask R-CNN

Используемая архитектура maskrcnn_resnet50_fpn основана на глубокой сверточной сети ResNet-50, в которую встроена пирамида признаков (Feature Pyramid Network, FPN) [10]. Последняя позволяет извлекать иерархические признаки с различной степенью детализации, что критически важно при работе с объектами разного масштаба.

Для текущей работы представляет интерес именно способность модели обрабатывать объекты с разнообразной формой, нечеткими границами и переменным размером, что характерно для нефтяных разливов на водной поверхности. Такие объекты могут иметь сложную, фрактальную структуру, сливающуюся с фоном, и находиться в условиях переменного освещения. Благодаря FPN и раздельному предсказанию масок для каждого объекта, Mask R-CNN позволяет выявлять подобные загрязнения с высокой пространственной точностью.

B. Swin V2 Tiny

Swin V2 Tiny принадлежит к семейству vision-трансформеров, в которых сверточные операции заменены механизмом самовнимания. Ключевая особенность модели — оконное (shifted-window) внимание: изображение разбивается на непересекающиеся окна фиксированного размера (4 × 4 пикселя), внутри которых вычисляется локальное внимание, затем окна сдвигаются, что позволяет передавать информацию между соседними областями без значительного роста вычислительной сложности [11]. Такая стратегия создает иерархию признаков, аналогичную пирамиде FPN в CNN, но строящуюся средствами трансформера.

Сеть начинается с патч-разделителя, переводящего входное изображение в последовательность векторов-«токенов» [12]. Далее следуют четыре стадийных блока, где последовательно уменьшается разрешение карты признаков и увеличивается ее глубина. На каждом уровне применяются слой нормализации, оконное внимание и небольшая двухслойная полносвязная проекция (MLP), которая действует как позиционный преобразователь признаков.

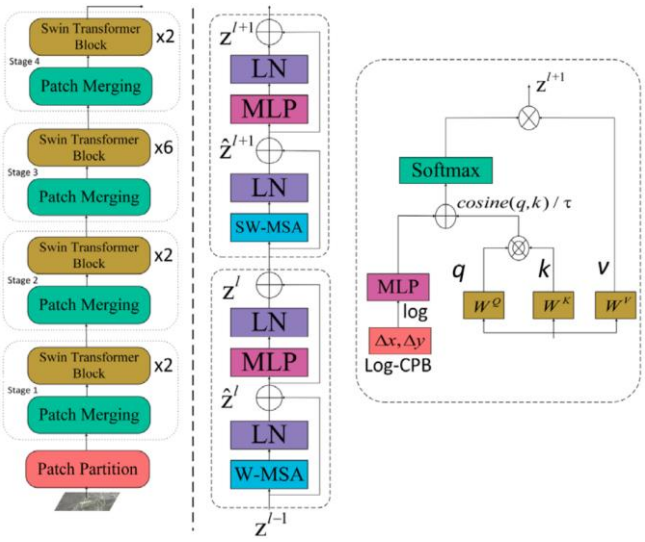


Рис. 3. Архитектура Swin V2 Tiny – слева, два последовательных блока Swin Transformer - справа

В контексте данной задачи ценен прежде всего механизм оконного самовнимания, позволяющий одновременно захватывать мелкие локальные детали нефтяной пленки и интегрировать их в глобальную картину сцены. Нефтяные разливы обладают неустойчивой текстурой, размытой границей и варьирующимся масштабом. Трансформерная иерархия признаков Swin, формируемая засчет последовательного сдвига окон, обеспечивает устойчивое выделение таких структур даже при сильных колебаниях освещенности и отражениях воды. Дополнительное агрегирование контекста между уровнями способствует различению тонких пленочных участков от «рябого» водного фона, что критично для раннего обнаружения загрязнений. Таким образом, Swin V2 Tiny предоставляет точные и коэнвариантные представления, делая ее перспективным трансформерным ориентиром для сравнения с классической Mask R-CNN в задачах экологического мониторинга.

V. Сравнение

Для оценки качества работы моделей сегментации нефтяных разливов были проведены количественные и визуальные сравнения двух нейросетевых архитектур: Mask R-CNN (ResNet-50 + FPN) и Swin V2 Tiny. Обе модели обучались на одних и тех же данных с аннотированными масками, содержащими три класса: oil (нефть), water (вода), others (прочее окружение). Таблица 1 отображает усредненные значения основных метрик по этим классам.

ТАБЛИЦА I. Оценка метрик

	Mask R-CNN	**Swin V2 Tiny**
Mean IoU	0.8395	0.7033
Mean Precision	0.9639	0.7700
Mean Recall	0.8674	0.8900
Mean F1-score	0.8913	0.8200

Модель Mask R-CNN продемонстрировала высокие значения по всем ключевым метрикам, что свидетельствует о ее способности выполнять стабильную и точную сегментацию. Особенно высоким оказалось значение Precision, указывающее на относительно небольшое количество ложноположительных предсказаний. При этом Recall также находится на высоком уровне, что говорит о тенденции модели к охвату значительной части целевых объектов в большинстве случаев. Эти результаты подтверждают, что Mask R-CNN уверенно справляется с задачей выделения классов в условиях разнородного визуального фона.

В свою очередь, Swin V2 Tiny продемонстрировал более скромные результаты. Трансформер хорошо справился с задачей обнаружения объектов (высокий Recall), но при этом допустил больше ложных срабатываний, что видно по более низкому Precision. Это особенно характерно для случаев с размытыми границами между «нефтью» и «прочими объектами», где контекст оказывается важнее пиксельной детализации.

ТАБЛИЦА II. Оценка метрик по классам

Класс	**Модель**	**IoU**	**Precision**	**Recall**	**F1-score**
Вода	Mask R-CNN	0.79	0.94	0.84	0.88
	Swin V2 Tiny	0.68	0.75	0.87	0.81
Нефть	Mask R-CNN	0.83	0.96	0.85	0.90
	Swin V2 Tiny	0.88	0.98	0.89	0.94
Прочее	Mask R-CNN	0.90	0.99	0.91	0.90
	Swin V2 Tiny	0.55	0.58	0.91	0.71

При сравнении метрик по каждому классу (таблица 2) выявляется различие в специализации моделей. Для класса «вода» Mask R-CNN достигла IoU = 0.79 и Precision = 0.94, в то время как Swin V2 — IoU = 0.68 и Precision = 0.75. Это говорит о лучшей способности сверточной архитектуры обрабатывать однородные участки водной поверхности с четкими границами.

Интересно, что на классе «нефть» Swin V2 показала более высокий IoU = 0.88 против 0.83 у Mask R-CNN и высокий F1-score = 0.94. Однако Precision в обоих случаях был на высоком уровне (0.96 и 0.98 соответственно), что подтверждает уверенность моделей в своих предсказаниях, хотя Mask R-CNN при этом точнее очерчивает формы.

По классу «прочее» (инфраструктура, берег, корабли) преимущество явно на стороне Mask R-CNN: IoU = 0.90 и Precision = 0.99 против 0.55 и 0.58 у Swin V2. Это объясняется тем, что трансформер испытывает трудности с детальной сегментацией сложных и разнородных объектов, тогда как CNN-архитектура с FPN эффективно обрабатывает мелкие детали и формы.

Для дополнения количественной оценки был проведен визуальный анализ. На рисунке 4 представлены результаты Mask R-CNN: предсказанные маски в большинстве случаев точно соответствуют эталонным. Видны четкие границы объектов и минимальное наложение классов.

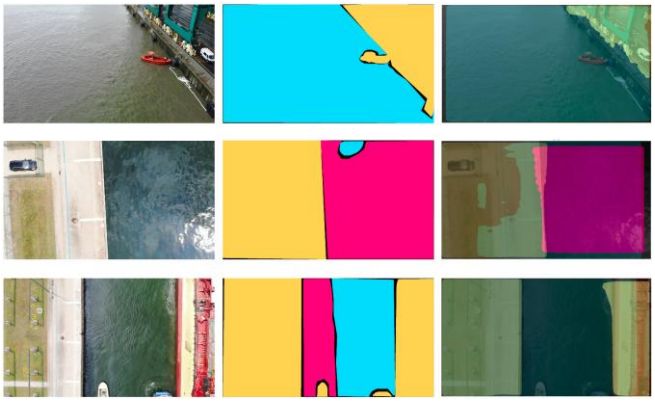


Рис. 4. Примеры результатов сегментации модели Mask R-CNN. Первый столбец – исходные изображения; второй – эталонные маски (разметка вручную); третий – предсказания модели. Маски отображаются в полупрозрачном виде: желтым цветом обозначены прочие объекты, голубым – водная поверхность, розовым – нефтяные разливы

На рисунке 5 показаны результаты работы Swin V2 Tiny: маски зеленого цвета (нефть) часто выходят за пределы реальных разливов, а фоны (желтый — прочее, синий — вода) подвержены переобобщению. При этом модель иногда успешно фиксирует зоны, где Mask R-CNN теряет уверенность — что подчеркивает различие в архитектурных стратегиях.

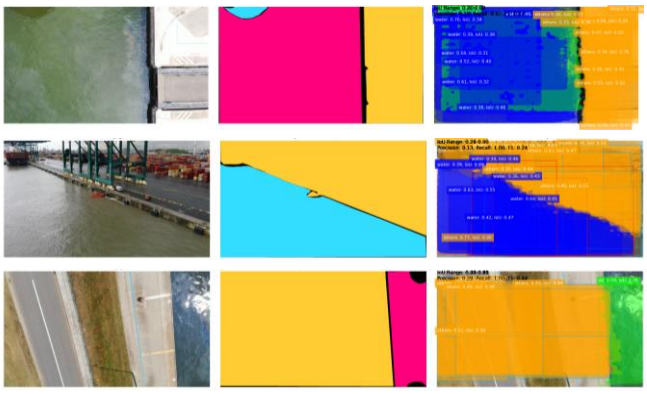


Рис. 5. Примеры результатов сегментации модели Swin V2 Tiny. Первый столбец — исходные изображения; второй – эталонные маски; третий – предсказания модели. Визуализация результата отличается цветовой схемой: желтым обозначены прочие объекты, синим – вода, зеленым – нефтяные разливы

Особенно заметным фактором, влияющим на работу обеих моделей, стала рябь на поверхности воды и блики, возникающие при съемке под разными углами. В случае Mask R-CNN такие участки иногда интерпретируются как граница между классами, что приводит к частичному искажению сегментации воды. У Swin V2 Tiny, напротив, рябь и отражения нередко ошибочно классифицируются как нефтяные разливы, что проявляется в виде избыточного распространения зеленых масок. Это указывает на чувствительность трансформера к локальным шумам текстуры и необходимость дополнительной постобработки или адаптивного порогирования при применении в реальных условиях.

Таким образом, можно заключить, что Mask R-CNN более надежна и сбалансирована в задаче сегментации всех трех классов, особенно в сложных урбанизированных сценах. Однако Swin V2 Tiny демонстрирует потенциал в захвате обширных зон загрязнений и может использоваться в качестве вспомогательного фильтра для предварительного выделения проблемных участков.

VI. Заключение

В данной работе рассмотрено применение двух современных нейросетевых архитектур — Mask R-CNN и Swin V2 Tiny — для сегментации нефтяных разливов на водной поверхности. Обе модели были обучены на одинаковом датасете с разметкой по трем классам: нефть, вода и прочее окружение. Проведенное сравнение показало, что Mask R-CNN обеспечивает более стабильную и точную сегментацию по большинству метрик, особенно в части локализации и распознавания сложных объектов инфраструктуры. В то же время Swin V2 Tiny продемонстрировал высокое значение Recall и точность на классе «нефть», что может быть полезно при разработке систем с приоритетом на чувствительность к загрязнениям.

Визуальный анализ подтвердил различия в поведении моделей в условиях ряби и неоднородности воды. Результаты показывают, что комбинированное или выборочное применение обеих архитектур может стать основой для эффективных систем автоматического экологического мониторинга. Перспективным направлением может стать использование ансамблей и постобработки с учетом физических особенностей сцены.

Литература

[1] He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 42, no. 2. P. 386-397. DOI: 10.1109/TPAMI.2018.2844175.

[2] Грищенко, Д. И. Классификация земного покрова и землепользования / Д. И. Грищенко // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 30-35. – EDN AYJWGA.

[3] Али, Б. Алгоритмы навигации беспилотных летательных аппаратов с использованием систем технического зрения / Б. Али, Р. Н. Садеков, В. В. Цодокова // Гироскопия и навигация. – 2022. – Т. 30, № 4(119). – С. 87-105. – DOI 10.17285/0869-7035.00105. – EDN ETCJST.

[4] Лим, В. Л. Исследование вопроса распознавания светофоров / В. Л. Лим // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : СБОРНИК СТАТЕЙ НАУЧНО-ТЕХНИЧЕСКОГО СЕМИНАРА СТУДЕНТОВ КАФЕДРЫ «ИНЖЕНЕРНОЙ КИБЕРНЕТИКИ», Москва, 30 декабря 2023 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2023. – С. 34-39. – EDN KDXQCK.

[5] Chen L.-C., Zhu Y., Papandreou G. et al. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2018. Vol. 40, no. 4. P. 834-848. DOI: 10.1109/TPAMI.2017.2699184.

[6] Altunian. Oil Spills [Электронный ресурс] // Hugging Face. URL: https://huggingface.co/datasets/altunian/oil_spills (дата обращения: 20.03.2025).

[7] He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 42, no. 2. P. 386-397. DOI: 10.1109/TPAMI.2018.2844175.

[8] He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 42, no. 2. P. 386-397. DOI: 10.1109/TPAMI.2018.2844175.

[9] Computer vision system: A tool for evaluating the quality of wheat in a grain tank / U. I. Minkin, A. V. Panchenko, A. Y. Shkanaev [et al.] // Proceedings of SPIE - The International Society for Optical Engineering, Vienna, 13–15 ноября 2017 года. Vol. 10696. – Vienna: SPIE, 2018. – P. 106961. – DOI 10.1117/12.2310100. – EDN XXEYIP.

[10] He K., Gkioxari G., Dollár P., Girshick R. Mask R-CNN // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017. Vol. 42, no. 2. P. 386-397. DOI: 10.1109/TPAMI.2018.2844175.

[11] Liu Z., Hu H., Lin Y. et al. Swin Transformer V2: Scaling Up Capacity and Resolution // arXiv preprint [Электронный ресурс]. 2022. arXiv:2111.09883. URL: https://arxiv.org/abs/2111.09883 (дата обращения: 16.04.2025).

[12] Wang W., Xie E., Li X. et al. Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions // IEEE/CVF International Conference on Computer Vision (ICCV). 2021. P. 548-558. DOI: 10.1109/ICCV48922.2021.00061.

ГРНТИ 28.23.37, 73.31.81, 55.30.31
ASJC 1702, 1706
OECD 1.02 EP

Применение синтетических данных из UnrealEngine для обучения модели сегментации мебели

С. Аскари Хеммат
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2214004@misis.ru

Аннотация— В данной статье рассматривается применение синтетических данных для обучения моделей сегментации объектов интерьера. С использованием движка Unreal Engine 5 был создан синтетический датасет сцен с различной мебелью, включая столы, стулья и диваны. Для повышения качества обучающей выборки вручную проведена разметка объектов с помощью инструмента COCO Annotator в формате COCO. Основной целью исследования стало сравнение эффективности моделей Fast R-CNN и RT-DETR при сегментации указанных классов мебели на основе синтетических данных. В статье приведены количественные оценки качества сегментации и скорости работы моделей, демонстрирующие потенциал применения синтетических сцен для обучения в задачах компьютерного зрения.

Ключевые слова — Сегментация объектов, Синтетические данные, Unreal Engine 5, COCO Annotator, Fast R-CNN, RT-DETR..

I. Введение

Современные задачи компьютерного зрения всё чаще решаются с применением методов глубинного обучения, для обучения которых требуется большое количество разметенных данных. Однако в ряде прикладных областей, таких как сегментация объектов интерьера, получение качественных размеченных датасетов в реальных условиях сопряжено с рядом трудностей. Сложности включают трудоёмкость и высокую стоимость ручной разметки, ограниченность разнообразия сцен, а также невозможность масштабируемого управления условиями съёмки. Всё это стимулирует интерес к использованию синтетических данных, создаваемых с помощью графических движков и специализированных инструментов визуализации. Одним из наиболее мощных решений в этой области является Unreal Engine 5 [1], предоставляющий фотореалистичную визуализацию, гибкую настройку сцен и точный контроль над условиями освещения и компоновкой объектов.

Создание синтетических датасетов открывает новые возможности для обучения и тестирования моделей компьютерного зрения в управляемых и воспроизводимых условиях. Использование трёхмерных сцен позволяет сгенерировать большое количество изображений с различными ракурсами, фонами и условиями освещения, что способствует повышению обобщающей способности моделей. Более того, синтетические данные позволяют легко масштабировать [2] объём обучающей выборки и оперативно адаптировать её под конкретную задачу, например, сегментацию ограниченного количества классов объектов.

В данной работе рассматривается задача сегментации мебели по синтетическим изображениям, созданным на движке Unreal Engine 5 [3]. В качестве объектов сегментации выбраны три типа предметов: стол, стул и диван. Для получения обучающего датасета были сгенерированы сцены с различным расположением мебели в интерьерах, а затем вручную размечены с использованием инструмента COCO Annotator. Полученные данные были сохранены в формате COCO, что обеспечивает совместимость с большинством современных фреймворков глубокого обучения.

Ключевыми вызовами при использовании синтетических данных для обучения моделей сегментации являются:

- Семантический разрыв между синтетическими и реальными изображениями, который может снижать точность моделей при применении их на реальных данных.
- Корректность и полнота разметки, особенно в случае ручной аннотации, даже если она проводится на сгенерированных изображениях.
- Выбор и настройка модели, способной эффективно обучаться на синтетических данных и сохранять высокую точность на реальных изображениях.

A. *Синтетические данные*

Для генерации сцен использовались готовые [4] 3D-модели мебели, размещённые в интерьерах с различной геометрией и освещением. Unreal Engine 5 позволил обеспечить фотореализм и разнообразие условий визуализации, включая дневной и ночной свет, различные углы обзора, плотность объектов и текстурные особенности. Каждое изображение сцены сохранялось [5] с высоким разрешением, после чего вручную размечалось в COCO Annotator.

На рисунке 1 представлены некоторые примеры синтетических изображений мебели, сгенерированных в Unreal Engine 5:

Рис. 1. Примеры изображений мебели, сгенерированных в Unreal Engine 5

B. Разметка данных

Для разметки использовался специализированный веб-инструмент COCO Annotator, поддерживающий формат аннотаций COCO, который широко применяется в задачах сегментации и детекции объектов..

Процесс аннотирования проводился вручную с использованием инструмента Magic Wand Tool, позволяющего полуавтоматически выделять объекты на изображении на основе схожести цветов и границ. Этот подход существенно ускорил процесс сегментации и обеспечил высокую точность выделения контуров мебели, особенно в случае объектов с чёткими границами и однородной текстурой.

На рисунке 2 представлены примеры разметки данных на COCO Annotator:

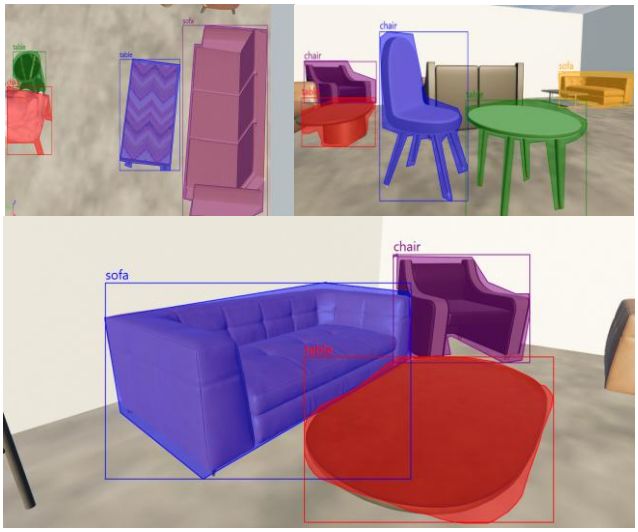


Рис. 2. Примеры разметки данных на COCO Annotator

Размеченный датасет, включающий изображения и аннотации в формате COCO, был опубликован в открытом доступе на платформе Hugging Face Datasets Hub,. Датасет может быть использован как для обучения, так и для тестирования и дообучения моделей сегментации в прикладных проектах и исследованиях.

II. Нейросетевые архитектуры

A. Fast R-CNN

YOLOv8n ast R-CNN (Fast Region-based Convolutional Neural Network) — это улучшенная версия оригинального алгоритма R-CNN, предложенная Р. Гиршиком в 2015 году, которая значительно повысила эффективность и точность распознавания объектов на изображениях. Основной задачей архитектуры является детекция объектов с последующей классификацией [6] и определением ограничивающих рамок (bounding boxes). Fast R-CNN устраняет несколько узких мест своего предшественника и представляет собой более быструю и интегрированную систему.

В отличие от R-CNN, где каждый предложенный регион (region proposal) обрабатывается отдельно через свёрточную нейронную сеть (CNN), Fast R-CNN выполняет единственное свёрточное проходное преобразование по всему изображению. На его основе формируется карта признаков (feature map), из которой с помощью алгоритма Region of Interest (RoI) Pooling извлекаются фиксированные признаки для каждого предложенного региона.

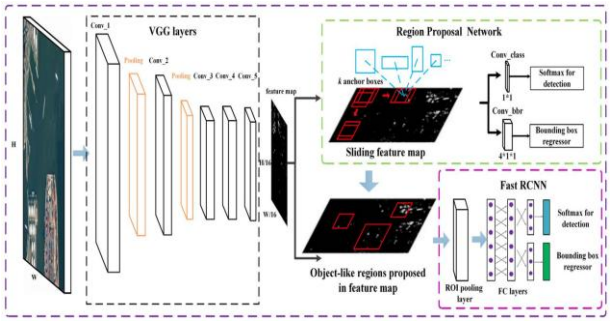


Рис. 3 Архитектура Fast R-CNN

После RoI Pooling извлечённые признаки передаются в полносвязные слои, которые параллельно решают две задачи: классификацию объектов (softmax) и регрессию ограничивающих рамок (bounding box regression). Такая архитектура позволила существенно ускорить процесс и повысить точность.

Ключевые преимущества Fast R-CNN:

- Скорость: благодаря общему проходу по изображению и RoI Pooling, сеть быстрее R-CNN в несколько раз.

- Совместное обучение: классификация и локализация выполняются одновременно, что обеспечивает лучшую сходимость модели.
- Улучшенное качество: более точное позиционирование объектов за счёт регрессии координат рамок.

Fast R-CNN часто используется как база для последующих улучшений, таких как Faster R-CNN и Mask R-CNN, которые добавляют модуль генерации регионов и сегментацию соответственно. Несмотря на появление более современных архитектур, Fast R-CNN остаётся важным этапом в развитии методов object detection.

B. *RT-DETR*

RT-DETR (Real-Time DEtection TRansformer) — это одна из первых архитектур, использующих механизм трансформеров для детекции объектов [7] в реальном времени [2]. Она представляет собой развитие идеи DETR (DEtection TRansformer), предложенной Facebook AI Research, но с упором на производительность и скорость, позволяющую использовать модель в высоконагруженных приложениях.

Классические CNN-архитектуры, такие как YOLO или R-CNN, хорошо справляются с локальными признаками, но ограничены в улавливании глобального контекста. RT-DETR, в свою очередь, использует механизм внимания (attention), [8] что даёт ей возможность эффективно анализировать как локальные, так и глобальные зависимости на изображении.

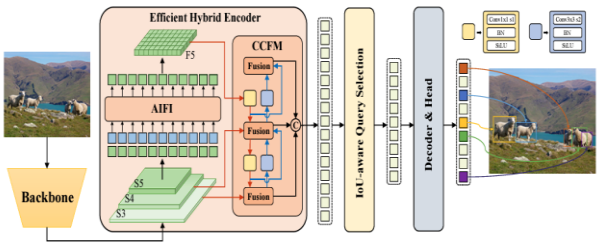


Рис. 4 Архитектура сети RT-DETR

Архитектура RT-DETR включает несколько ключевых компонентов:

- **Backbone**: модуль для извлечения признаков, основанный на ResNet, ConvNeXt или Swin Transformer.
- **Transformer encoder-decoder**: основной блок, преобразующий признаки изображения в предсказания о расположении и классе объектов.
- **Object queries**: фиксированное количество запросов, каждый из которых "ищет" один объект на изображении.
- **Feed-forward head**: окончательный слой, который выдаёт координаты рамок и классы объектов.

Одним из ключевых преимуществ RT-DETR [9] является **отказ от NMS (Non-Maximum Suppression)**. В классических архитектурах используется NMS для устранения дублирующихся предсказаний, но он увеличивает задержку. RT-DETR предсказывает фиксированное число объектов, что делает дополнительную фильтрацию ненужной. Это повышает стабильность вывода и уменьшает вычислительную нагрузку [10].

C. *Метрики оценки качества*

Для оценки качества моделей детекции объектов широко используются метрики [11] Precision, Recall, mAP50. Каждая из них позволяет оценить различные аспекты производительности моделей.

Precision — это метрика, которая показывает долю правильно определённых объектов среди всех предсказанных моделью объектов. Высокое значение Precision свидетельствует о том, что модель редко делает ошибки в виде ложных срабатываний.

Recall — это метрика, измеряющая способность модели находить все объекты на изображении. Высокое значение Recall указывает на то, что модель эффективно обнаруживает объекты, даже если они сложноразличимы.

mAP50 (средняя точность при IoU ≥ 50%) показывает, насколько точно модель распознаёт объекты, соответствующие эталонным рамкам, с перекрытием не менее 50%. Эта метрика позволяет оценить, насколько хорошо модель локализует объекты, не требуя слишком высокой точности.

III. Сравнение

A. *Метрики качества*

Результаты обучения можно обобщить в следующей таблице:

Модель	Precision	Recall	mAP50
Fast-RCNN	0.91	1.0	0.95
RT-DETR	0.94	1.0	0.98

Табл. 1 Метрики качества моделей

Таблица демонстрирует, что обе модели показывают высокие результаты по всем основным метрикам качества. RT-DETR немного превосходит Fast R-CNN по точности и mAP@50. Учитывая при этом архитектурные особенности RT-DETR и её оптимизацию под задачи реального времени [12], данная модель может быть предпочтительнее в системах с ограниченным временем отклика, несмотря на незначительное различие в качестве.

B. Результаты детекций

Теперь посмотрим примеры обработки изображения модели на тестовых данных и также на реальных изоб-

ражениях, чтобы проверить работоспособность модели вне синтетических данных:

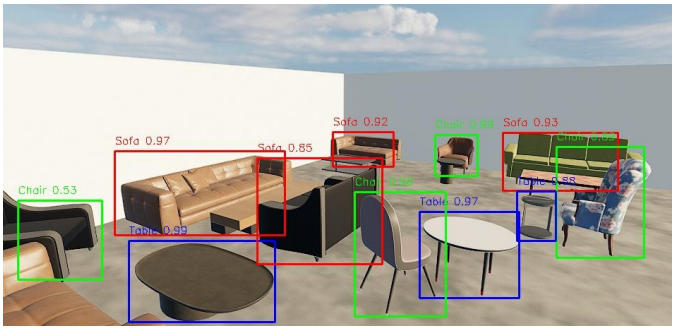


Рис. 5 Пример на синтетических данных

Рис. 6 Пример на реальном изображении

Рис. 7 Пример на реальном изображении

IV. Заключение

В данной работе рассматривалась задача сегментации мебели с использованием синтетических данных, полученных в виртуальной среде Unreal Engine. Для создания обучающего датасета использовались 3D-сцены с фотореалистичной визуализацией помещений, в которых автоматически производилась аннотация объектов с помощью инструмента Magic Wand в COCO Annotator. Разметка была выложена в открытый доступ на платформе HuggingFace [13], что обеспечивает воспроизводимость и доступность результатов.

В рамках исследования была проведена оценка производительности моделей Fast R-CNN и RT-DETR, обученных на сгенерированных синтетических данных. Обе модели продемонстрировали высокие показатели по метрикам Precision, Recall и mAP50, что подтверждает применимость синтетических сцен в задачах сегментации.

Модель RT-DETR показала наивысшие значения по метрикам что делает её предпочтительным выбором для внедрения в реальные приложения.

Литература

[1] Feltrin, Leonardo, et al. "Synthetic datasets for deep learning in indoor environments: A survey." *Sensors* 21.12 (2021): 3997.

[2] С. А. Зайцева. Генерация синтетических данных для задач компьютерного зрения: методы и перспективы. // Сборник научных трудов «Информационные технологии и интеллектуальные системы» (2023).

[3] Wren, H. E., and Y. D. Kim. "Using Unreal Engine for photorealistic synthetic data generation in machine learning." *Journal of Imaging* 7.3 (2021): 49.

[4] Chowdhury, Samiul Haque, and M. Bennamoun. "Using game engines for training deep learning models: A review." *arXiv preprint* arXiv:2207.00539 (2022).

[5] Rozantsev, Artem, Vincent Lepetit, and Pascal Fua. "On rendering synthetic images for training an object detector." Computer Vision and Image Understanding 137 (2015): 24–37.

[6] Marion, Vincent, et al. "Synthesizing training data for object detection in indoor scenes." arXiv preprint arXiv:1801.01293 (2018).

[7] Stojanov, Stefan, Kevin Zhang, and S. Song. "Sim2Real Transfer for Indoor Scene Understanding using Synthetic RGB-D Data." Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2022, pp. 151–160.

[8] Singh, A., and P. K. Gupta. "Fast R-CNN based segmentation for indoor object detection." *International Journal of Computer Applications* 182.20 (2018): 1–5.

[9] Liu, Zhaoyang, Yuxin Fang, and Xizhou Zhu. "RT-DETR: Real-time detection transformer." *arXiv preprint* arXiv:2304.08069 (2023).

[10] Е.А. Ашманова, И.А. Ширеторова. Нейросетевые методы идентификации конкретного представителя семейства кошачьих, сборник статей на тему «Искусственный Интеллект в Промышленных, Коммерческих, Медицинских и Финансовых Приложениях» (2024).

[11] Chowdhury, Samiul Haque, and M. Bennamoun. "Using game engines for training deep learning models: A review." *arXiv preprint* arXiv:2207.00539 (2022).

[12] И.М. Бахвалов, С.В. Старцев. Распознавание БПЛА различных классов средствами компьютерного зрения , сборник статей на тему «Искусственный Интеллект в Промышленных, Коммерческих, Медицинских и Финансовых Приложениях» (2024).

[13] HuggingFace Furniture Detection Dataset: https://huggingface.co/datasets/sina09/UnrealEngine_Furniture (2025).

ГРНТИ 20.53.19, 28.23.37
ASJC 1707
OECD 1.02 EP

Нейросетевые методы для распознавания дронов и птиц в воздушном пространстве

Е. А. Ашманова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2005713@edu.misis.ru

С. В. Старцев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2401133@edu.misis.ru

Аннотация — В работе рассматривается задача детекции и классификации дронов и птиц в воздушном пространстве. Задача имеет важное значение для обеспечения безопасности аэропортов, охраны объектов и экологического мониторинга. Для решения задачи применялись современные SOTA-модели - YOLOv8L и RT-DETR Large. Для обучения и тестирования был собран и размечен датасет, составленный из изображений, взятых из открытых источников и отражающих различные условия съемки. Результатом статьи является сравнение метрик качества дообученых моделей для решения задачи распознавания и классификации объектов в воздухе.

Ключевые слова — Компьютерное зрение, Детекция объектов, Классификация объектов в воздухе, Распознавание дронов, Распознавание птиц, Беспилотные летательные аппараты (БПЛА), YOLO, RT-DETR.

I. Введение

В последние годы задачи распознавания и классификации объектов в воздушном пространстве приобрели особую актуальность в связи с ростом использования беспилотных летательных аппаратов (БПЛА) или, по-другому, дронов [1, 2, 3]. В этом контексте методы компьютерного зрения выступают эффективным инструментом для обнаружения таких объектов. Современные алгоритмы обработки изображений и машинного обучения позволяют системам анализировать визуальные данные - фотографии и видеопотоки - и распознавать беспилотники по их характерным визуальным признакам, таким как форма, размер и особенности движения [4].

Одновременно с задачей распознавания дронов становится актуальным потребность в идентификации БПЛА от природных объектов, в частности птиц. Для этого современные системы мониторинга требуют разработки высокоточных и быстрых алгоритмов, способных надежно обнаруживать и классифицировать объекты в различных условиях съёмки. Это особенно важно для обеспечения безопасности аэропортов, охраны стратегически значимых объектов и экологического контроля [5].

Технологии глубокого обучения и нейросетевые модели, такие как YOLO и DETR, демонстрируют значительный прогресс в области компьютерного зрения и успешно применяются для задач детекции и классификации объектов на изображениях и видео. YOLO (You Only Look Once) [6] отличается высокой скоростью обработки и эффективностью, что делает её предпочтительным выбором для приложений с ограниченными вычислительными ресурсами и требованиями к оперативности [7]. В то же время модели на основе трансформеров, такие как RT-DETR [8] и его усовершенствованные версии, обеспечивают высокую точность и лучшее понимание контекста сцены за счёт механизма внимания, что особенно важно при сложных условиях наблюдения [9].

Однако для успешного применения этих моделей в задачах распознавания дронов и птиц требуется учёт множества факторов, включая разнообразие условий освещения, погодные влияния, а также особенности форм и движений объектов. Кроме того, ограниченность и неоднородность существующих датасетов усложняет обучение универсальных моделей, способных работать в реальных сценариях. В связи с этим актуальным является создание и использование разнообразных размеченных наборов данных, отражающих реальные условия воздушного пространства.

II. Наборы данных

Для решения задачи детектирования и классификации объектов в воздухе был собран, размечен и опубликован пользовательский датасет [10], состоящий из изображений, взятых из открытых источников. Датасет включает 1 466 изображений, для каждого их которых имеется аннотация в виде ограничивающих прямоугольников, точно локализирующих объекты и относящих их к одному из двух классов: «Птица» или «Дрон».

Для оптимальной оценки качества моделей данные были разделены на тренировочную, валидационную и тестовую выборки в пропорциях 70%, 20% и 10% соответственно, что соответствует современным стандартам машинного обучения и позволяет эффективно контролировать процесс обучения и обобщающую способность моделей.

Средний размер изображений составляет 0.41 мегапикселя, с медианным разрешением 640×640 пикселей, что обеспечивает однородность входных данных и упрощает этапы предобработки. Размеры объектов на изображениях варьируются от мелких, находящихся на заднем плане, до крупных, расположенных вблизи камеры. Такое разнообразие масштабов и дистанций повышает устойчивость модели к различным условиям съемки.

Общее количество аннотаций достигает 8657, что в среднем соответствует примерно шести объектам на изображение. Высокая плотность объектов на

изображениях отражает реалистичные сценарии, где воздушные объекты часто присутствуют в группах или скоплениях, что усложняет задачу детекции и требует от моделей способности к точному разделению и локализации множества объектов в одном кадре.

На рисунках 1-4 изображены примеры рассматриваемых данных в реальных условиях.

Рис. 1. Пример изображения с объектами из двух разных классов

Рис. 2. Пример изображения с объектом из класса «Дрон»

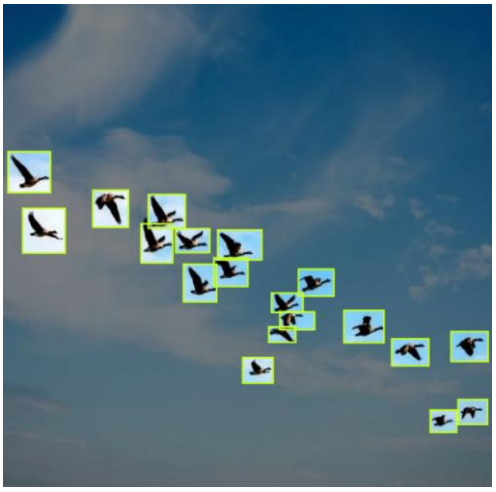


Рис. 3. Пример сцены с множеством объектов из одного класса «Птица»

Рис. 4. Пример кадра с объектом класса «Дрон» на заднем плане

III. Нейросетевые архитектуры

A. YOLOv8 (You Only Look Once).

YOLOv8 — это последняя версия популярной нейросетевой архитектуры для детекции объектов в реальном времени, которая продолжает развитие оригинальной модели YOLO. Эта модель сочетает в себе улучшенную точность и увеличенную скорость обнаружения, при этом сохраняя ключевые преимущества предыдущих версий – высокую производительность и универсальность применения. YOLOv8 разработана как единый фреймворк, поддерживающий множество задач компьютерного зрения, включая обнаружение, сегментацию и классификацию объектов, что делает её удобной и эффективной для широкого спектра приложений.

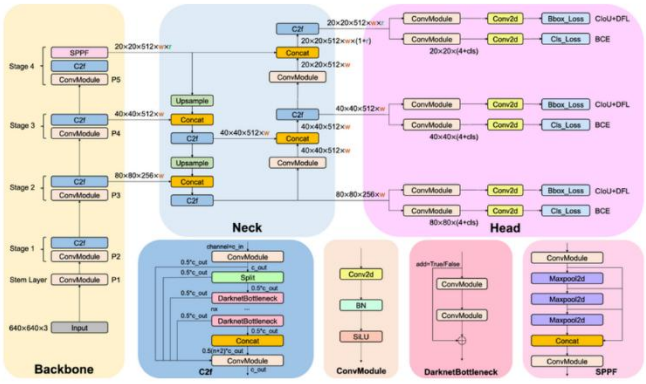


Рис. 5. Схематичное изображение архитектуры YOLOv8

Архитектура YOLOv8, как и её предшественники, построена по принципу одноступенчатого детектора, что означает одновременное выполнение задач локализации и классификации объектов за один проход нейросети. В этой версии используется более глубокая и усовершенствованная сеть, что позволяет повысить точность распознавания при сохранении высокой скорости обработки.

Основные элемент архитектуры YOLOv8 включают:

- Backbone (основной блок):

  Модифицированная сеть CSPDarknet, основанная на Darknet с внедрением CSPNet (Cross-Stage

Partial Network), которая оптимизирует извлечение признаков, повышая производительность и снижая вычислительные затраты [11].

- Neck (средний слой):

  PANet (Path Aggregation Network): Сеть использует PANet для улучшенной агрегации информации с разных уровней признаков. Это позволяет YOLOv8 работать лучше с объектами разного масштаба.

  FPN (Feature Pyramid Network): Эта структура улучшает точность распознавания мелких объектов за счет комбинирования признаков с разных уровней сети [12].

- Head (выходной слой):

  Формирует предсказания ограничивающих рамок, классов объектов и их точность. Для повышения качества результатов применяются методы оценки перекрытия рамок IoU (Intersection over Union) и алгоритм подавления лишних предсказаний Non-Maximum Suppression (NMS) [13].

YOLOv8 включает в себя несколько усовершенствований, которые помогают улучшить точность и производительность модели. Одним из них является оптимизация вычислений. YOLOv8 использует более компактные и быстрые архитектуры для обработки изображений, что позволяет ускорить обучение и детекцию. Также модель использует многомасштабное обучение, т.е. она обучается с учётом объектов различных размеров, что улучшает её способность работать с малыми и крупными объектами на изображении.

B. YOLOv12

YOLOv12 представляет собой новое поколение архитектуры серии YOLO, которое значительно отличается от YOLOv8 как по структуре, так и по принципам работы. В основе YOLOv12 лежит внимание, что выводит модель за рамки классических сверточных нейронных сетей, применяемых в YOLOv8. Это позволяет более эффективно выделять значимые области изображения и улучшать качество детекции объектов, особенно в сложных сценах.

Главное отличие YOLOv12 — использование усовершенствованного основного блока - Residual Efficient Layer Aggregation Network (R-ELAN) с глубокими остаточными связями и разделяемыми свёртками, что повышает способность модели извлекать признаки с меньшими вычислительными затратами. В то время как YOLOv8 опирается на модифицированную CSPDarknet, YOLOv12 предлагает более эффективное и глубокое представление данных, что положительно сказывается на точности и скорости работы.

Средний слой (neck) в YOLOv12 также претерпел значительные изменения: внедрён механизм Area Attention с ускорением FlashAttention, который позволяет модели фокусироваться на ключевых зонах изображения, снижая при этом нагрузку на память и ускоряя обработку. Это улучшает распознавание объектов разных размеров и повышает устойчивость к визуальным шумам, чего нет в классическом PANet и FPN, используемых в YOLOv8.

Выходной слой (head) YOLOv12 оптимизирован для более точного предсказания ограничивающих рамок и классов объектов за счёт расширенного рецептивного поля и новых функций активации, таких как SiLU. Также применяются специализированные функции потерь, которые лучше балансируют задачи локализации и классификации, что улучшает качество детекции по сравнению с YOLOv8.

Кроме архитектурных новшеств, YOLOv12 предлагает расширенную масштабируемость — четыре варианта модели с разным соотношением скорости и точности, а также улучшенную поддержку квантования и pruning, что делает её более пригодной для использования на edge-устройствах с ограниченными ресурсами. В сравнении с YOLOv8 это позволяет добиться прироста точности при одновременном снижении задержек и уменьшении размера модели

C. RT-DETR (Real-Time DEtection TRansformers)

RT-DETR – это современный детектор объектов в реальном времени, разработанный компанией Baidu и основанный на концепции DETR, но с рядом важных усовершенствований для повышения скорости и точности. В основе модели лежит гибридный кодировщик, который эффективно обрабатывает признаки изображения с разных масштабов, разделяя внутримасштабное взаимодействие признаков (AIFI) и межмасштабное слияние (CCFM). Это позволяет модели лучше учитывать разнообразие размеров объектов и снижать вычислительные затраты.

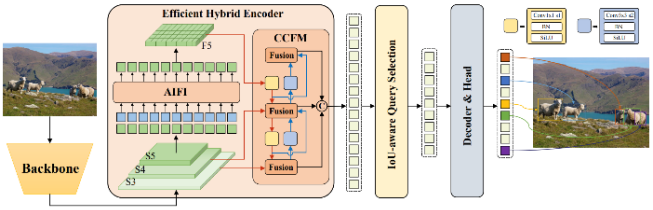


Рис. 6. Архитектура RT-DETR

Входными данными для кодировщика служат признаки с трёх последних этапов основной свёрточной сети (обычно уровни S3, S4, S5), что обеспечивает богатое представление изображения. Особенностью RT-DETR является использование механизма выбора запросов с учётом IoU (IoU-aware query selection), который помогает модели сосредоточиться на наиболее релевантных объектах, улучшая качество детекции.

Декодер модели итеративно оптимизирует объектные запросы, предсказывая ограничивающие рамки и оценки уверенности без необходимости в традиционной постобработке типа не максимального подавления (NMS). Такая архитектура позволяет RT-DETR работать эффективно и стабильно в режиме реального времени, обеспечивая гибкую настройку скорости вывода за счёт использования различных слоёв декодера без переобучения.

Кроме того, RT-DETR базируется на трансформерах зрения (Vision Transformer, ViT), что даёт модели

возможность улавливать глобальный контекст изображения и улучшать качество детекции, особенно в сложных сценах с множеством объектов и разнообразным фоном. Благодаря этому RT-DETR превосходит многие классические детекторы по точности локализации и устойчивости к вариативности данных, при этом сохраняя высокую производительность на ускоренных платформах, таких как CUDA с TensorRT.

Таким образом, RT-DETR сочетает в себе преимущества трансформерной архитектуры и эффективного гибридного кодировщика, обеспечивая высокую точность и скорость обнаружения объектов в реальном времени без сложных этапов постобработки.

IV. Сравнение

Для решения задачи детекции и классификации объектов в воздушном пространстве были выбраны три модели из SOTA-подхода: YOLOv8, YOLOv12 и RT-DETR. Обучение каждой из моделей проводилось в течение 200 эпох. Для оптимизации использовался оптимизатор AdamW с начальными гиперпараметрами: learning rate (lr) - 0.01, weight_decay - 5e-4, что соответствует рекомендуемым значениям по умолчанию для соответствующих архитектур.

Для оценки качества работы моделей использовались метрики Precision, Recall и mAP@50. Метрика Precision отражает точность модели, показывая, какую долю из всех предсказанных объектов составляют действительно корректные обнаружения. Другими словами, она характеризует, насколько верны предсказания модели. Метрика Recall оценивает полноту модели, то есть способность находить все объекты интересующего класса на изображениях, показывая, какую часть от общего числа реальных объектов модель смогла обнаружить. Метрика mAP@50 (mean Average Precision при пороге IoU 0,5) объединяет информацию о точности и полноте, предоставляя комплексную оценку качества детекции объектов, учитывая степень совпадения предсказанных и истинных ограничивающих прямоугольников.

Для более наглядного понимания метрик введем следующие обозначения:

- TP (True Positive) – модель верно обнаружила объект нужного класса (дрон или птица).
- FP (False Positive) – модель ошибочно классифицировала объект другого класса или фон как целевой объект.
- FN (False Negative) – модель не обнаружила объект нужного класса, хотя он присутствовал на изображении [16,17].

Стоит отметить, что TN (True Negative) в задачах детекции обычно не применяется, так как она отражает количество правильно отвергнутых фонов, что не всегда релевантно для оценки качества детекции.

На основе этих величин рассчитываются основные метрики:

- $Precision = \frac{TP}{TP+FP}$ – показывает, какую долю всех предсказанных объектов модель определила корректно;
- $Recall = \frac{TP}{TP+FN}$ – отражает, какую часть всехприсутствующих объектов модель смогла обнаружить.

В таблице 1 показаны количественные характеристики используемых подходов [12, 13]. Анализ представленных результатов показывает, что модель YOLOv8 демонстрирует наивысшее значение Precision (84,3%), что свидетельствует о её высокой точности при обнаружении объектов. В то же время модель YOLOv12 превосходит YOLOv8 и RT-DETR по показателю Recall (82,0%), указывая на более полное выявление объектов на изображениях. При этом по метрике mAP@50, отражающей точность локализации объектов при заданном пороге IoU, лидирует RT-DETR с результатом 88,6%, тогда как YOLOv12 занимает промежуточное положение (84,6%), а YOLOv8 — 81,9%. Таким образом, YOLOv8 обеспечивает наиболее точное обнаружение, YOLOv12 — более полное, а RT-DETR выделяется высокой точностью определения границ объектов.

ТАБЛИЦА I. Оценка детектирующей части

	YOLOv8	**YOLO v12**	**RT-DETR**
Precision	84.3%	83.3%	83.6%
Recall	81.0%	82.0%	79.7%
mAP@50	81.9%	84.6%	88.6%

На рисунке 7 показаны графики для модели YOLOv8, демонстрирующие изменение ключевых метрик обучения и валидации модели от числа эпох. На данных графиках видно, что присутствует стабильное снижение функций потерь как на тренировочных, так и на валидационных данных, что свидетельствует о корректном процессе обучения. Метрики качества же заметно быстро достигают высоких значений и остаются стабильными на протяжении большей части обучения, что указывает на эффективное выявление и классификацию объектов.

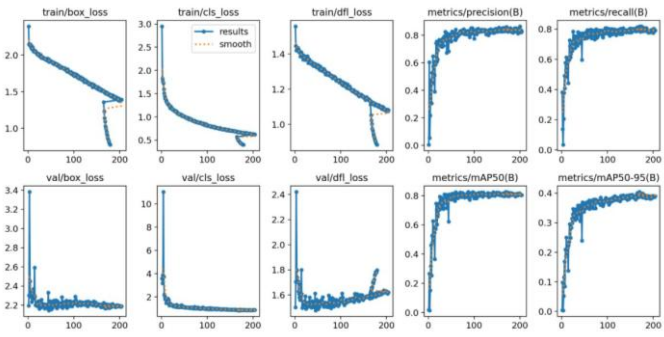
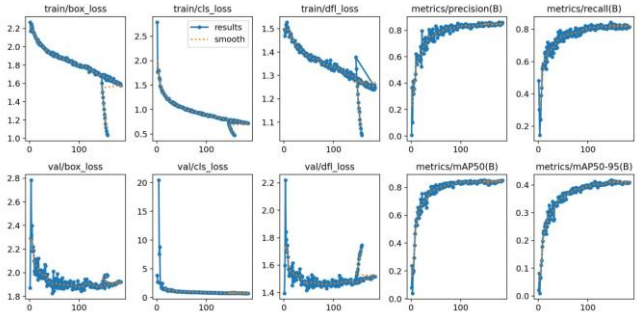


Рис. 7. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели YOLOv8

На рисунке 8 приведены графики обучения модели YOLOv12. Данные для 8 и 12 версии YOLO отличаются

незначительно. В отличии от 8 версии 12 оказалась более стабильна в метриках качества. На графиках наглядно видно, что эти графики имеют намного меньше выбросов. Это свидетельствует о большей устойчивости и предсказуемости процесса обучения модели.

Рис. 8. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели YOLOv12



Аналогичный анализ был проведен для модели RT-DETR. Результаты представлены также в виде графиков на рисунке 9. На графиках, отражающих процесс обучения модели RT-DETR, также наблюдается устойчивое снижение всех функций потерь по мере увеличения числа эпох. В отличие от графиков обучения YOLOv8, на графиках для RT-DETR наблюдается более плавная динамика уменьшения функций потерь как на тренировочных, так и на валидационных данных. Это говорит о более размеренном и устойчивом процессе обучения модели.

Кроме того, для RT-DETR характерно меньше выбросов и колебаний на графиках метрик качества, что свидетельствует о стабильности обучения и высокой устойчивости модели к внутренней вариативности датасета. Также такой характер графиков указывает на то, что модель RT-DETR менее подвержена переобучению или случайным ошибкам на отдельных эпохах.

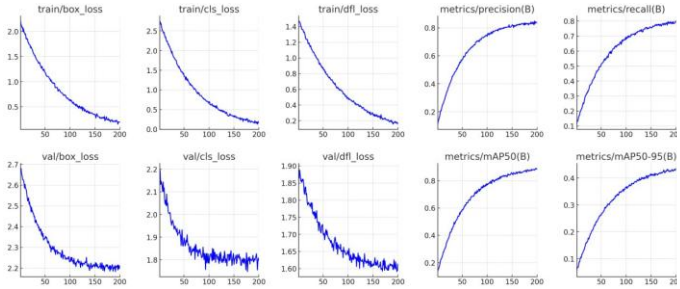


Рис. 9. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели RT-DETR

Таким образом, RT-DETR демонстрирует более сглаженное и стабильное обучение, что может быть преимуществом при работе с реальными, разнообразными данными в задачах, где важна предсказуемость поведения модели на новых примерах.

Экспериментальные результаты показали, что все три модели демонстрируют высокие показатели качества, но у каждой модели есть свои сильные стороны. YOLOv8 обеспечивает наивысшее значение Precision (84,3%), что говорит о её высокой точности обнаружения объектов, а YOLOv12 превосходит остальные по Recall (82,0%), обеспечивая более полное выявление объектов на изображениях. RT-DETR, в свою очередь, достигает более высокого значения mAP@50 (88,6%), что свидетельствует о её превосходстве в точной локализации объектов. Графики обучения показывают, что обе модели YOLO быстрее достигают высоких значений метрик, а RT-DETR отличается более плавным и стабильным процессом обучения с меньшим количеством выбросов.

Таким образом, выбор оптимальной модели зависит от конкретных требований задачи: если приоритетом является высокая точность и скорость обнаружения, предпочтительнее YOLOv8; если важна максимальная полнота выявления объектов — YOLOv12; а для задач, где критична точная локализация и устойчивость обучения, лучше подходит RT-DETR.

Результаты работы дообученых нейронных сетей YOLO и RT-DETR представлены на рисунках 10-15.

Рис. 10. Результат работы дообученной нейронной сети YOLOv8

Рис. 11. Результат работы дообученной нейронной сети YOLOv8

Рис. 12. Результат работы дообученной нейронной сети YOLOv12

Рис. 13. Результат работы дообученной нейронной сети YOLOv12

Рис. 14. Результат работы дообученной нейронной сети RT-DETR

Рис. 15. Результат работы дообученной нейронной сети RT-DETR

V. Заключение

В данной работе были рассмотрены и проанализированы две архитектуры нейронных сетей из SOTA-подхода —YOLOv8, YOLOv12 и RT-DETR, применяемые для задачи детекции и классификации объектов в воздухе– дронов и птиц. Для обучения и тестирования моделей был собран и размечен пользовательский датасет, включающий 1466 изображений с аннотациями двух классов - «Дрон» и «Птица», отражающих разнообразные условия съёмки. Оценка качества работы моделей проводилась с использованием стандартных метрик компьютерного зрения: Precision, Recall и mAP@50.

Результаты работы показывают, что все три модели эффективно решают задачи детекции и классификации объектов в воздухе, но с разными сильными сторонами. Модель YOLOv8 лучше справляется с классификацией объектов, обеспечивая высокую точность, в то время как RT-DETR выигрывает в более точной локализации. Модель YOLOv12 выигрывает в вопросе полноты обнаружения и сбалансированное качество.

Таким образом, если приоритетом является максимально точное и сбалансированное обнаружение объектов в реальном времени, предпочтение следует отдать YOLOv8 или YOLOv12 в зависимости от требований к полноте выявления. Если же задача требует максимальной точности локализации и устойчивости к вариативности данных, более подходящим выбором будет RT-DETR. Полученные результаты могут быть полезны при выборе оптимальной архитектуры для практических задач мониторинга воздушного пространства и обеспечения безопасности

Литература

[1] 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

[2] Bakhvalov I.M., Startsev S.V. Recognition of UAVs of Different Classes Using Computer Vision Methods // Computer Tools in Education. – 2024. – No. 2. – P. 34–40. – Available

at: http://www.sadekov.su/Articles/kik_2024_2.pdf (Accessed: 10 May 2025)

[3] Матяш, Д. С. Детекция беспилотных летательных аппаратов на фотографиях с использованием методов компьютерного зрения / Д. С. Матяш // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 92-99. – EDN COFLJI.

[4] Pazychev, Dmitry & Bakulev, K. & Sadekov, Rinat. (2023). Low-Cost Navigation System for UAV. 1-6. 10.23919/ICINS51816.2023.10168469

[5] Kassaba M., Abu Zitar R., ElFallah Seghrouchni A., Barbaresco F. Bird/Drone Detection and Classification using Classical and Deep Learning Methods [Electronic resource] / M. Kassaba, R. Abu Zitar, A. ElFallah Seghrouchni, F. Barbaresco. – 2023. – Available at: https://www.researchgate.net/publication/369868096_BirdDrone_Detection_and_Classification_using_Classical_and_Deep_Learning_Methods (Accessed: 10 May 2025)

[6] Ultralytics. YOLOv8: State-of-the-Art Object Detection Model [Electronic resource]. – GitHub repository. – 2023. – Available at: https://github.com/ultralytics/ultralytics (Accessed: 10 May 2025)

[7] Redmon J., Divvala S., Girshick R., Farhadi A. You Only Look Once: Unified, Real-Time Object Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2016. – P. 779–788.

[8] RT-DETR: Real-Time Detection Transformer [Electronic resource]. – GitHub repository. – 2024. – Available at: https://github.com/lyuwenyu/RT-DETR (Accessed: 10 May 2025)

[9] Carion N., Massa F., Synnaeve G., Usunier N., Kirillov A., Zagoruyko S. End-to-End Object Detection with Transformers // European Conference on Computer Vision (ECCV). – 2020. – P. 213–229.

[10] Ashmanova E., Startsev S. Drons_and_Birds_Detections [Electronic resource] // Hugging Face. URL: eashmanova/Drons_and_Birds_Detection · Datasets at Hugging Face (Accessed: 10 May 2025)

[11] Huang J, Chen K., and Liu Z., (2017). Speed/accuracy trade-offs for modern convolutional object detectors. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/1611.10012 (Accessed: 10 May 2025).

[12] J. Redmon, S. Divvala, R. Girshick, R and A. Farhadi, (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/15 06.02640(Accessed: 10 May 2025).

[13] Redmon J.,Farhadi A., YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/1612.08242 (Accessed: 10 May 2025).

[14] Towards Data Science, "Confusion Matrix and Performance Metrics," Towards Data Science Blog, https://towardsdatascience.com/an introduction-to-performance-metrics-in-machine-learning543bfa9256b1 (Accessed: Accessed: 10 May 2025

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

[16] zantsev A., Lepetit V., Fua P. Flying Objects Detection from a Single Moving Camera // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2017. – Vol. 39, No. 5. – P. 879–892.

[17] Kellenberger B., Marcos D., Tuia D. Detecting Mammals in UAV Images: Best Practices to Address a Substantially Imbalanced Dataset with Deep Learning // Remote Sensing of Environment. – 2018. – Vol. 216. – P. 139–153.

ГРНТИ 28.23.37
ASJC 1707
OECD 1.02.EP

Использование подходов детектирования и оптического распознавания символов в задаче перевода формул в текстовый формат

В. В. Ащепкова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2411823@edu.misis.ru

Г. С. Листратенков
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2415186@edu.misis.ru

***Аннотация* — в рамках данной работы проводится сравнительный анализ комбинированного подхода к решению задачи автоматического распознавания и транскрипции формул химических реакций из изображений и PDF-документов. Представленная система объединяет методы детектирования объектов с использованием моделей YOLOv12 и RF-DETR для локализации областей с формулами, и Mathematical OCR (оптического распознавания символов) с применением нейросетей Nougat и TrOCR для последующей транскрипции обнаруженных формул в формат LaTeX. Эксперименты проводились на различных наборах научных документов и формул с целью оценки производительности моделей в условиях разной сложности формул. Она включала анализ точности обнаружения и корректности генерируемого LaTeX-кода, что позволило оценить эффективность и определить перспективы данного подхода для автоматизации обработки научных текстов и извлечения знаний из химических выражений.**

Ключевые слова — Компьютерное зрение, Глубокое обучение, Распознавание формул химических реакций, Детектирование, YOLOv12б RF-DETR, Nougat, TrOCR, LaTeX, mAP, IoU, Автоматическая обработка текста, Анализ научных документов.

I. Введение

Распознавание и транскрипция химических формул в научных документах является важной задачей, входящей в раздел Mathematical OCR, которая направлена на автоматизацию обработки, хранения и анализа научных знаний. Автоматическое извлечение выражений из изображений и PDF-документов позволяет значительно повысить эффективность научных исследований, упростить обмен информацией и облегчить создание интерактивных учебных материалов. В последние годы многие университеты СПИИРАН [1], университет Ватерлоо [2], Пекинский университет [3], исследовательские центры и компании (Mathpix) активно разрабатывают технологии автоматического распознавания формул. На международных конференциях, посвященных обработке текстов и компьютерному зрению: ICDAR (Международная конференция по анализу и распознаванию документов) [4], DAS (Системы анализа документов) [5], обсуждаются новые методы и алгоритмы, которые могут быть применены для автоматического извлечения выражений.

Ключевой задачей при создании таких систем является точное и надежное распознавание формул на основе изображений. Для решения этой задачи активно используются методы компьютерного зрения, в частности подходы, основанные на глубоком обучении. Среди них можно выделить два основных направления: детектирование объектов [6] и Mathematical OCR [7]. Детектирование объектов позволяет локализовать области с формулами на изображении, определяя их положение и границы, в то время как Mathematical OCR предоставляет возможность транскрибировать формулы в машиночитаемый формат, такой как LaTeX [8], обеспечивая возможность их дальнейшего редактирования и анализа.

Современные модели глубокого обучения, такие как YOLOv12 [9], RF-DETR [10] и Nougat [11], TrOCR [12], зарекомендовали себя как эффективные инструменты для решения задач детектирования и распознавания в области анализа документов. YOLOv12 и RF-DETR представляют собой передовые архитектуры для детектирования объектов, обеспечивающие высокую скорость и точность обнаружения. Nougat и TrOCR, в свою очередь, являются специализированными моделями для Mathematical OCR, способными транскрибировать сложные выражения в формат LaTeX.

Для обучения моделей глубокого обучения требуются большие объемы данных, что может быть вызовом при работе с математическими и химическими формулами, особенно редкими и сложными. Однако благодаря доступности отсканированных учебников и онлайн-библиотек научных публикаций, таких как arXiv 10 [13], задача обучения моделей становится более выполнимой.

В данной работе проводится сравнительный анализ подхода, основанного на детекции объектов и Mathematical OCR, в задаче распознавания и транскрипции формул химических реакций из изображений и PDF-документов. Исследование направлено на оценку применимости этих методов в различных условиях и анализ их эффективности в реальных сценариях автоматической обработки научных текстов.

II. Наборы данных

Для проведения анализа эффективности нейронных сетей в задаче детектирования и распознавания химических уравнений, был подготовлен авторский

датасет, состоящий их двух частей. Первая предназначена для задачи детекции химических уравнений на изображении. Вторая предназначена для оптического распознавания. Общий объем датасета – 2000 экземпляров, по 1000 для задач. Датасет опубликован в открытом доступе [14].

A. *Detection*

Датасет представляет собой обширную коллекцию из 1000 изображений страниц статей и книг по химии, на которых изображены уравнения химических реакций. Аннотация уравнений осуществлена при помощи прямоугольных ограничительных рамок (bbox).

Изображения были получены из открытых источников в сети интернет. Источники разнообразны, от учебников и самоучителей для школьников и студентов, до научных статей.

Данные содержат изображения в формате «png», в разрешении 640 на 640 пикселей. Для каждого изображения подготовлена аннотация, представляющая собой прямоугольную ограничительную рамку для каждого уравнения. Аннотации хранятся в формате «json» для каждого изображения.

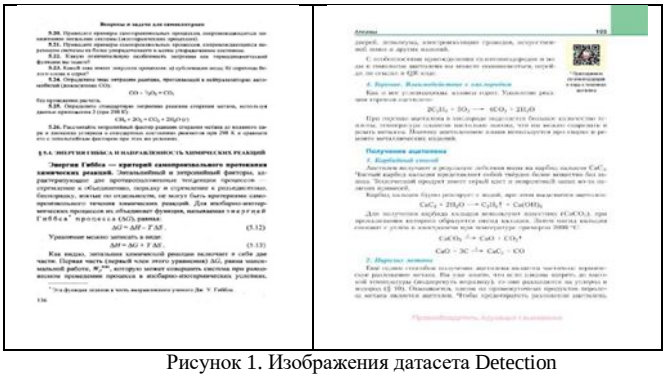


Рисунок 1. Изображения датасета Detection

B. *Chemical OCR*

Данный датасет предназначен для оптического распознавания уравнений химических реакций. Изображения получены из открытых источников в сети интернет. Всего 1000 экземпляров.

Все изображения имеют одинаковые размеры 751 на 128 пикселей. Для каждого подготовлена аннотация, представляющая собой текстовое описание уравнения химической реакции в формате LaTex. Аннотации хранятся как в «txt», так и в «json» формате. Ниже приведены примеры изображений:

$Zn + 2H_2O \rightarrow Zn_2 + H_2 \uparrow$	$C_6H_5CHCOOH + HCl \rightarrow C_6H_5CHCl - +COOH$
$2Cm + 6H_2O \rightarrow 2Cm_3 + 3H_2 \uparrow$	$C_6H_5CHCOOH + H_2 \rightarrow C_6H_5CH_2COOH + H_2O$

Рисунок 2. Изображения датасета Chemical OCR

III. Подготовка данных

Для проведения эксперимента данные из обоих датасетов были приведены в единый формат и размер: — 640x640 пикселей для задачи детекции, 751x128 пикселей для задачи оптического распознавания, а также созданы аннотации: Bounding Box для задачи детектирования и текстовое LaTex описание для задачи оптического распознавания. Данные были поделены на тренировочную и валидационную выборки.

Отдельно для расширения набора данных для задачи детектирования было решено применить следующую аугментацию:

- горизонтальный поворот;
- вертикальный поворот;
- размытие.

Для обучения нейросетей на датасете Detection, разметка была проведена посредством электронного сервиса CVAT.ai. Примеры разметки изображений приведены на рисунке 3.

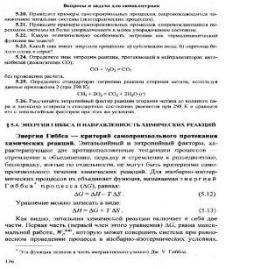
а	б
	

Рисунок 3 Разметка изображения а) Исходное б) С границами

Для обучения нейросетей на датасете Chemical OCR, аннотации были созданы вручную. Примеры аннотаций приведены на рисунке 4.

а	б
$Zn + 2H_2O \rightarrow Zn_2 + H_2 \uparrow$	Zn ^ { + } 2 H _ { 2 } O \rightarrow Zn _ { 2 } + H _ { 2 } \uparrow
$2Cm + 6H_2O \rightarrow 2Cm_3 + 3H_2 \uparrow$	2 Cm ^ { + } 6 H _ { 2 } O \rightarrow 2 Cm _ { 3 } + 3 H _ { 2 } \uparrow .

Рисунок 4 Аннотации к изображениям датасета Chemical OCR а) изображение б) LaTex описание

Для проведения тестирования эффективности нейронных сетей из датасетов были выделены тестовые выборки из случайных изображений.

IV. Нейросетевые архитектуры

A. *YOLOv12*

Архитектура YOLOv12 разработана для преодоления ограничений, связанных с механизмами внимания в задачах детектирования объектов в реальном времени. Для этого YOLOv12 включает в себя три ключевых усовершенствования: модуль Area Attention (A2), сети Residual Efficient Layer Aggregation Networks (R-ELAN) и оптимизации, связанные с архитектурой.

Модуль Area Attention (A2) предназначен для снижения вычислительной сложности, присущей традиционным механизмам внимания, A2 делит карту признаков на сегменты, сохраняя при этом большое поле восприятия. Это позволяет модели поддерживать широкий угол обзора, повышая скорость и эффективность.

Residual Efficient Layer Aggregation Networks (R-ELAN) (см. рис. 5) решает проблемы оптимизации,

возникающие при использовании механизмов внимания, путем внедрения остаточных соединений и методов масштабирования на уровне блоков, обеспечивая стабильное обучение. Кроме того, R-ELAN имеет переработанный метод агрегации признаков, который повышает производительность и эффективность.

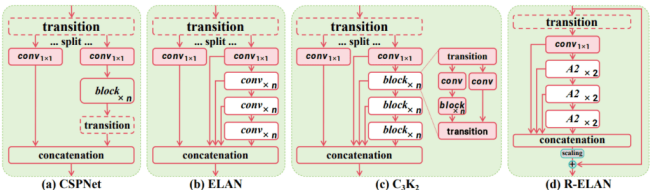


Рисунок 5. R-ELAN

B. *RF-DETR*

RF-DETR использует архитектуру LW-DETR с бэкбоном DINOv2, обученным на большом количестве данных. Это позволяет модели легко адаптироваться к новым задачам и доменам (например, подводным изображениям).

В Encoder используется vanilla ViT (Vision Transformer), где изображение разбивается на патчи, а затем обрабатывается трансформерными слоями. Для повышения скорости используется окно-селективное внимание (windowed self-attention), которое снижает вычислительную сложность. Информация агрегируется с нескольких уровней, чтобы создать мощные признаковые карты (feature maps).

Projector связывает энкодер и декодер. Использует C2f блок из YOLOv8 для повышения эффективности обработки признаков. Для больших версий RF-DETR проектор выводит мультискейловые признаки (с двух уровней разрешения), используя два параллельных C2f-блока. Один работает с 1/8 разрешения (деконволюция). Другой — с 1/32 (свертка со страйдом)

Decoder состоит из 3 трансформерных слоёв (в отличие от 6 у стандартного DETR), что снижает задержку. Использует деформируемое перекрёстное внимание (deformable cross-attention), что делает его быстрее и эффективнее на малых объектах.

Вводит гибридный механизм запросов (queries):

- Content queries — обучаемые эмбеддинги (как в DETR);
- Spatial queries — основаны на top-K признаках из Projector и трансформированы в пространственные эмбеддинги.

Бэкбон DINOv2 обеспечивает обобщающую способность модели на новые визуальные домены.

C. *Nougat*

Nougat (Neural Optical Understanding for Documents with Attentive Graphs) — это модель, разработанная для конвертации научных PDF-документов в структурированные текстовые представления. В основе архитектуры лежат принципы NAC.

Neural Attentive Circuits (NACs) [15] — это универсальная нейросетевая архитектура, сочетающая гибкость моделей общего назначения (например, Perceiver) с модульными индуктивными смещениями. NACs состоят из множества слабо и избирательно взаимодействующих модулей, которые обмениваются сообщениями по графу соединений, обучаемому или формируемому динамически. Архитектура включает два основных компонента:

Circuit Generator формирует дизайн схемы — сигнатуры, которые определяют связь между модулями, и коды, которые формируют вычисления, выполняемые модулями. Генератор может быть условным или безусловным.

Исполнитель схемы (Circuit Executor): принимает входные данные и дизайн схемы, выполняет токенизацию входа, итерирует состояния модулей через слой распространения (пропагаторы), и извлекает выход через модуль чтения (read-out).

Основные компоненты:

- ModFC и ModFFN: модули, где вычисления модулируются кодом;
- SKMDPA: механизм разреженного внимания, при котором вероятность взаимодействия между модулями зависит от расстояния между их сигнатурами;
- регуляризация графа: накладывает структурные приоритеты (например, масштабируемость или кластеризация), чтобы избежать полного соединения всех модулей.

D. *TrOCR*

TrOCR представляет собой систему оптического распознавания текста, построенную на основе архитектуры Transformer encoder-decoder. Модель разделена на два основных компонента: Encoder и Decoder.

В Encoder входное изображение масштабируется до фиксированных размеров и разделяется на регулярную сетку непересекающихся патчей, что соответствует подходу Vision Transformer (ViT). Патчи линеаризуются, проецируются в эмбеддинговое пространство и дополняются позиционными эмбеддингами. При инициализации от DeiT используется также distillation token. Полученная последовательность подаётся на вход Transformer-энкодеру, обеспечивающему извлечение глобальных и локальных визуальных признаков.

Decoder реализован на базе стандартной Transformer decoder-архитектуры и включает модули masked self-attention и encoder-decoder attention. В процессе генерации декодер принимает скрытые представления из энкодера в качестве ключей и значений, а также собственные предыдущие предсказания в качестве запросов. Автокорреляция по временной оси маскируется для предотвращения утечки информации о будущих токенах.

V. Обучение и результаты

A. *YOLOv12*

Для обучения была выбрана модель YOLOv12m. В качестве набора данных использовался датасет Detection.

При параметрах:

- количество эпох — 30;

- размер батча — 4;
- оптимизатор — Адам;
- lr — 0,003;
- patience — 10;
- размер картинок — 640.

Во время обучения удалось достичь следующих результатов, представлены на рисунке 6 и в таблице 1.

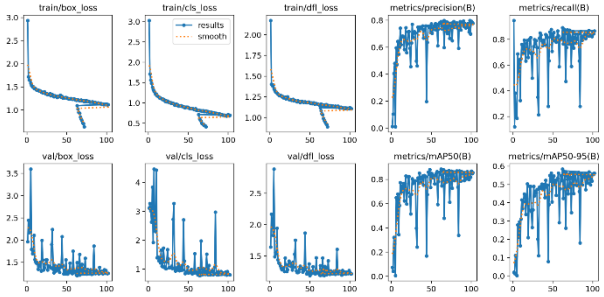


Рисунок 6. Метрики обучения

R	mAP50	mAP50-95
0,892	0,886	0,665

Таблица 1. Результаты обучения YOLOv12

Функции потерь демонстрируют снижение на протяжении обучения, что указывает на корректную сходимость модели, несмотря на разбросы. Снижение всех типов потерь подтверждает, что модель успешно оптимизирует предсказания.

Показатели точности (Precision) и полноты (Recall) растут. Это свидетельствует о способности модели правильно классифицировать большинство предсказанных объектов.

Значения mAP50 и mAP50–95 также демонстрируют положительную динамику на протяжении обучения. Это указывает на высокую точность детекции при умеренных порогах IoU и разумную устойчивость модели при более строгих условиях, что подтверждает её пригодность для задач локализации формул с разной степенью сложности.

Результаты применения показали, что модель может успешно определять границы формул, однако возможны неточности: пропуск формулы, наслоение их областей (см. рис. 7).

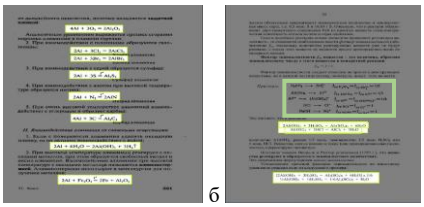

а б

Рисунок 7. Результаты YOLOv12 на тестовых данных

B. RF-DETR

Второй в эксперименте обучалась модель RF-DETR. В качестве набора данных был выбран датасет Detection.

При параметрах:

- количество эпох — 30;
- размер батча — 4;
- оптимизатор — Адам;
- lr — 0,00001;
- lr_backbone — 0,00001;
- размер картинок — 640.

Результаты, полученные по завершении обучения представлены на рисунке 7 и в таблице 2.

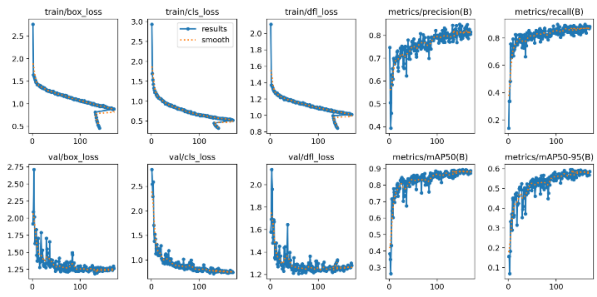


Рисунок 7. Метрики обучения

R	mAP50	mAP50-95
0,883	0,889	0,679

Таблица 2. Результаты обучения RF-DETR

Функции потерь на обучающей и валидационной выборках демонстрируют стабильное и постепенное снижение на протяжении всех эпох, что свидетельствует о корректной сходимости модели и сбалансированной оптимизации параметров. Модель улучшается в пространственной локализации и в предсказаниях.

Метрики точности (Precision) и полноты (Recall) демонстрируют устойчивый рост, что подтверждает способность модели не только правильно классифицировать найденные объекты, но и обнаруживать их в значительном большинстве случаев.

Показатели mAP50 и mAP50–95 отражают высокое качество детекции при стандартном пороге IoU и достаточную устойчивость при более строгих условиях оценки.

На тестовых данных видно, что модель успешно определяет границы формул, возможны неточности детекции: пропуск формулы (см. рис. 8).

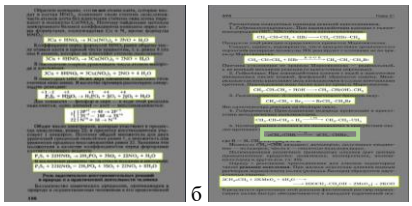

а б

Рисунок 8. Результаты RF-DETR на тестовых данных

C. Nougat

Для обучения была выбрана модель Nougat. В качестве набора данных был выбран датасет Chemical OCR.

При параметрах:

- max_length — 128;

- num_beams — 1;
- learning_rate — 0,00001;
- per_device_train_batch_size — 8;
- per_device_eval_batch_size — 4;
- num_train_epochs — 15.

Также был изменен стандартный BPE-токенизатор:

Спецтокены	LaTeX-символы (\rightarrow, \leftarrow и др.)
Нормализация	Приведение к нижнему регистру, удаление акцентов, декомпозиция Unicode (NFD)
Предтокенизация по пробелам	Более стабильное разбиение формул
Ограничение словаря до 5000 токенов	Контроль охвата и размера модели.

Таблица 3. Описание токенизатора

Прогресс обучения представлен на рисунке 9 и в таблице 4.

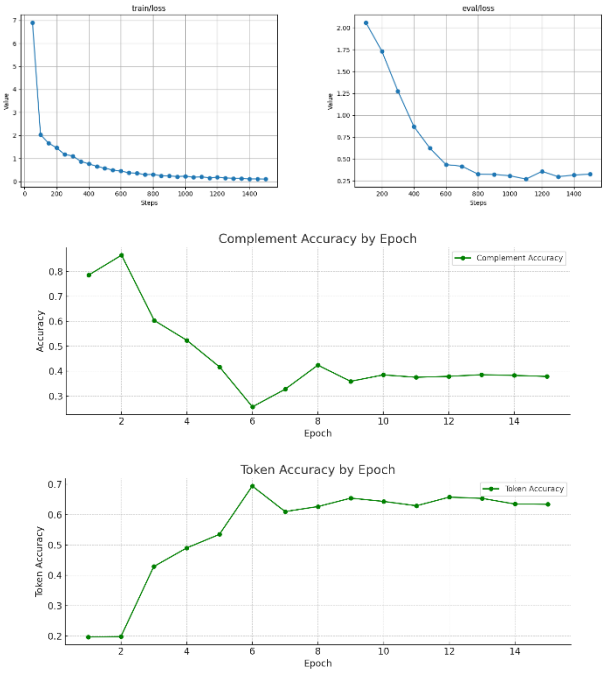


Рисунок 9. Метрики обучения

На графике тренировочных потерь (train/loss) наблюдается быстрая экспоненциальная сходимость тренировочных потерь в начале обучения. После начального резкого снижения (до примерно 300-400 шагов) темп снижения потерь замедляется, стремясь к асимптотическому уровню ближе к нулю. Это указывает на то, что модель эффективно усваивает паттерны из тренировочных данных.

Валидационные потери (eval/loss): Валидационные потери также демонстрируют тенденцию к снижению, хотя и менее выраженную, чем в тренировочных данных. Минимальное значение валидационных потерь достигается примерно к 1200 шагам, после чего наблюдается незначительное увеличение (возможно, свидетельствующее о начале переобучения).

Результаты применения к тестовым данным представлены на рисунке 10. Формулы переведены точно, за исключением спецсимволов.

а	$Mg + H_2SO_4 \longrightarrow MgSO_4 + H_2$	Mg+H_{2}SO_{4}-MgSO_{4}+H_{2}
б	$Na + Cl_2 \longrightarrow NaCl$	Na+Cl_{2}-NaCl

Рисунок 10. Результаты Nougat на тестовых данных

D. TrOCR

Для обучения была выбрана модель TrOCR («microsoft/trocr-base-stage1»). В качестве набора данных был выбран датасет Chemical OCR.

При параметрах:

- max_length — 128;
- num_beams — 4;
- learning_rate — 0,00001;
- per_device_train_batch_size — 4;
- per_device_eval_batch_size — 2;
- num_train_epochs — 10.

Использовался донастроенный токенизатор (см. табл. 3). Результаты обучения представлены на рисунке 11 и в таблице 5.

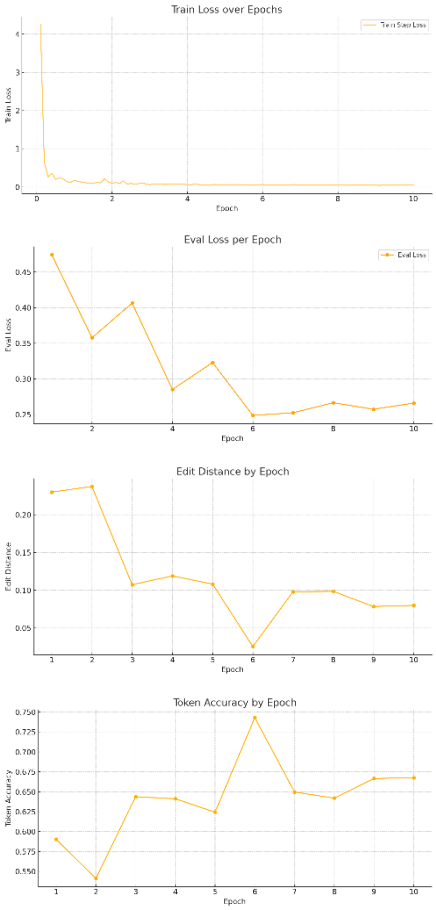


Рисунок 11. Метрики обучения

График Train Loss демонстрирует быструю и экспоненциальную сходимость в начале обучения. Наблюдается резкое снижение потерь в первых эпохах,

после чего темп снижения замедляется, стремясь к асимптотическому уровню, близкому к нулю. Это указывает на то, что модель эффективно усваивает паттерны из тренировочных данных.

График Eval Loss также демонстрирует тенденцию к снижению, хотя и с большей волатильностью по сравнению с Train Loss. Минимальное значение Eval Loss достигается примерно к эпохе 6, после чего наблюдается стабилизация и незначительное увеличение потерь.

График демонстрирует тенденцию к снижению Edit Distance (расстояние редактирования, также известное как расстояние Левенштейна) по мере увеличения числа эпох обучения.

Edit Distance является метрикой, измеряющей минимальное количество операций (вставки, удаления, замены), необходимых для преобразования одной строки в другую. В контексте OCR, снижение Edit Distance указывает на улучшение точности распознавания текста моделью.

Результаты применения к тестовым данным представлены на рисунке 12. В обоих случаях модель показывает точные результаты.

а	$2Mg + O_2 = 2MgO$	2Mg + O_{2} = 2MgO
б	$2NaBH_4 + 2H_2O \rightarrow B_2H_6 + 2NaOH + 2H_2$	2NaBH_{4} + 2H_{2}O \\\\rightarrow B_{2}H_{6} + 2NaOH + 2H_{2}

Рисунок 12. Результаты TrOCR на тестовых данных

Исходя из результатов эксперимента можно сделать вывод, что модели успешно решают задачи, для которых они были созданы. YOLOv12 и RF-DETR достигли сопоставимых метрик mAP и recall, эффективно определяя границы объектов. Nougat и TrOCR продемонстрировали хорошее качество распознавания, особенно TrOCR — с наименьшей ошибкой (Edit Distance).

VI. Заключение

В процессе исследования был выполнен анализ нескольких подходов для решения задачи извлечения формул химических реакций из pdf-документов и изображений. Нами были подготовлены два датасета для проведения обучения нейронных сетей.

В качестве модели для детекции были выбраны YOLOv12 и RF-DETR, для оптического распознавания — Nougat и TrOCR. Для каждой из моделей были рассмотрены: архитектура, параметры, процесс обучения, тип данных и формат аннотаций. Проводилось обучение на соответствующих датасетах, анализировались метрики и проверялась работоспособность модели на тестовых данных.

На основании результатов обучения установлено, что RF-DETR незначительно превосходит YOLOv12 в задаче детекции формул на странице. По показателю R, mAP50 примерно одинаковы, по показателю Precision опережает конкурента на 5,7%.

В задаче оптического распознавания на небольшом датасете лучше себя показала модель TrOCR. Nougat сложнее в настройке и обучении по причине необходимости использования собственных токенизаторов, тонкой настройки дополнительных параметров. По показателям точности и скорости обучения модели примерно равны.

Используя обученные нами модели RF-DETR и TrOCR, был реализован сервис по автоматическому извлечению формул из pdf-документов и изображений и дальнейшему распознаванию в LaTex-формат.

Мы считаем, что для дальнейшего повышения эффективности моделей для поставленной в исследовании задачи необходимо расширение датасетов и кратное увеличение вычислительных мощностей.

VII. Литература

[1] Применение преобразования Хафа и метода наименьших квадратов для распознавания математических формул. Available at:https://www.mathnet.ru/links/237b4e99a1f2a353c45abb3b9682852f/trspy420.pdf (Accessed: May 31, 2025).

[2] A Fuzzy Logic Approach to Handwriting Recognition of Mathematical Expressions. Available at: https://www.scg.uwaterloo.ca/mathbrush/publications/fuzzyTechReport2010.pdf (Accessed: May 31, 2025).

[3] ConvMath: A Convolutional Sequence Network for Mathematical Expression Recognition. Available at: https://www.researchgate.net/publication/347797202_ConvMath_A_Convolutional_Sequence_Network_for_Mathematical_Expression_Recognition (Accessed: May 31, 2025).

[4] ICDAR 2024: International Conference on Document Analysis and Recognition. Available at: https://icdar2024.net/ (Accessed: May 31, 2025).

[5] Document Analysis Systems – Springer Conference Series. Available at: https://link.springer.com/conference/das (Accessed: May 31, 2025).

[6] Дедов, А. Д. Обнаружение кораблей на спутниковых изображениях с использованием компьютерного зрения / А. Д. Дедов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики», Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет «МИСИС», 2024. – С. 36–41. – EDN RVELMU.

[7] Mathematics – OCR Qualification Information. Available at: https://www.ocr.org.uk/subjects/mathematics/ (Accessed: May 31, 2025).

[8] The LaTeX Project – Official Site. Available at: https://www.latex-project.org/ (Accessed: May 31, 2025).

[9] YOLOv12 Models Documentation – Ultralytics. Available at: https://docs.ultralytics.com/ru/models/yolo12/#key-features (Accessed: May 31, 2025).

[10] RF-DETR: Object Detection with RF-Transformer. Available at: https://blog.roboflow.com/rf-detr/ (Accessed: May 31, 2025).

[11] Blecher, F., Bishop, T., Rossmann, J., & Fink, M. (2023). Nougat: Neural Optical Understanding for Academic Documents. arXiv preprint, arXiv:2310.02103. Available at: https://arxiv.org/abs/2310.02103 (Accessed: May 31, 2025).

[12] TrOCR: Transformer-based OCR with Pre-trained Vision and Language Models. Available at: https://arxiv.org/abs/2109.10282 (Accessed: May 31, 2025).

[13] ArXiv-10 Dataset – Papers with Code. Available at: https://paperswithcode.com/dataset/arxiv-10 (Accessed: May 31, 2025).

[14] CV_Chemical_reactions. Available at: https://huggingface.co/datasets/nxhxl/CV_Chemical_reactions (Accessed: May 31, 2025).

[15] Nougat Project Page – Meta AI. Available at: https://facebookresearch.github.io/nougat/ (Accessed: May 31, 2025).

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Исследование возможности детектирования дорожных знаков на основе нейросетевой модели YOLO

Ф. Е. Базалеев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2314593@edu.misis.ru

Е. И. Пиховская
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2409987@edu.misis.ru

Аннотация — в данной статье проводится сравнительный анализ трёх современных нейросетевых моделей — YOLOv12, YOLOv8 и RF-DETR — для задачи детектирования дорожных знаков, связанных с железнодорожными переездами. Обучение моделей проводилось на уникальном наборе данных, включающем три типа автодорожных знаков, размеченных вручную в системе CVAT. Для оценки моделей использовались стандартные метрики точности и скорости обработки. Проведённый анализ показал, что RF-DETR демонстрирует высокую точность детекции, в то время как YOLOv8 обеспечивает оптимальный баланс между скоростью и качеством. YOLOv12 проявляет конкурентоспособные результаты, но уступает в обработке в реальном времени. Полученные результаты могут быть полезны при разработке систем автономного вождения и интеллектуальных систем помощи водителям.

Ключевые слова — детектирование, распознавание дорожных знаков, нейронные сети, железнодорожный переезд, YOLO, RF-DETR.

I. Введение

Современное развитие интеллектуальных транспортных систем и автономных автомобилей требует высокой точности и надёжности при распознавании объектов дорожной инфраструктуры. Одним из критически важных элементов являются дорожные знаки, особенно те, что сигнализируют о приближении к зонам повышенной опасности, где требуется своевременное принятие решений со стороны водителя или системы автопилота.

Детекция дорожных знаков представляет собой сложную задачу компьютерного зрения, обусловленную разнообразием форм, цветов, условий освещённости и помех в виде фона или погодных факторов. Для её решения активно применяются нейросетевые модели, способные обеспечивать как высокую точность, так и обработку изображений в реальном времени. Среди таких моделей особенно выделяются архитектуры семейства YOLO (You Only Look Once)[1], отличающиеся скоростью работы, и трансформерные подходы, такие как RF-DETR, демонстрирующие прогресс в точности локализации и распознавания.

Статья посвящена сравнительному анализу современных моделей для задач детекции объектов: YOLOv12, YOLOv8 и RF-DETR. Основное внимание уделено их применимости в контексте распознавания дорожных знаков, связанных с железнодорожными переездами. В работе использовался специализированный датасет, размеченный вручную в системе CVAT, содержащий изображения трёх типов знаков. Оценка эффективности моделей проводилась с использованием стандартных метрик качества (точность, полнота, mAP) и производительности (скорость обработки кадров).

Целью данного исследования является определение оптимального решения для интеграции в интеллектуальные транспортные системы, что позволит повысить безопасность и эффективность управления дорожным движением в зонах железнодорожных переездов.

II. Наборы данных

Для обучения моделей был сформирован специализированный датасет, включающий изображения из двух различных источников, что обеспечило как разнообразие, так и релевантность данных в контексте задачи распознавания знаков, связанных с железнодорожными переездами. В него вошли следующие классы:

- 1.1 – железнодорожный переезд со шлагбаумом;
- 1.2 – железнодорожный переезд без шлагбаума;
- 1.3.1 – однопутная железная дорога.

Источники датасета:

Первым источником стал открытый набор данных RTSD, разработанный для задач детекции и классификации дорожных знаков в условиях, характерных для России. Он содержит изображения, снятые при различных погодных и световых условиях, что делает его ценным ресурсом для обучения моделей, ориентированных на реальные дорожные сценарии. Из RTSD было отобрано 400 изображений, однако исходная разметка оказалась неполной и неточной для поставленной задачи. В связи с этим все изображения были повторно размечены вручную в инструменте CVAT.

Рис. 1. Примеры размеченных кадров RTSD

Включение RTSD позволяет учитывать влияние неблагоприятных условий, таких как ограниченная видимость, плотная застройка, плохое качество дорожного покрытия и узкие дороги что критически важно для задач автономного вождения [2].

Рис. 2. Примеры кадров из датасета RTSD, снятых в сложных условиях обстановки

Второй источник – изображения, собранные вручную в виде скриншотов с онлайн-карт. Было собрано 623 изображения, содержащих участки дорог с железнодорожными переездами. Эти изображения также были размечены вручную в CVAT, что позволило расширить датасет за счёт редких и нестандартных ситуаций, слабо представленных в открытых источниках. Такой подход обеспечил адаптацию модели к условиям, типичным для российских регионов.

Особое внимание уделялось включению сложных случаев – например, частичное перекрытие знаков, плотное их размещение или наличие нескольких классов в кадре. Это должно повысить устойчивость модели при применении в реальных условиях.

Рис. 3. Примеры снимков с большим количеством знаков

В результате объединения двух источников был сформирован датасет из 1023 изображений. Для обучения моделей данные были разделены в пропорции 70:15:15 на обучающую, валидационную и тестовую выборки. Это обеспечило как эффективное обучение, так и независимую проверку качества работы моделей.

Была также обеспечена сбалансированность классов: каждое изображение содержит минимум один из трёх целевых знаков, что позволило достичь равномерного представления каждого класса. Перед обучением проведена валидация качества аннотаций, чтобы минимизировать ошибки разметки и повысить точность финальных моделей.

III. Нейросетевые архитектуры

Для решения задачи детекции дорожных знаков, связанных с железнодорожными переездами, были выбраны модели YOLOv8, YOLOv12 и RF-DETR. Эти архитектуры представляют собой современные решения в области компьютерного зрения и обеспечивают высокую точность и производительность в задачах локализации и классификации объектов. Их выбор обусловлен следующими причинами:

- YOLOv8 – одна из последних стабильных реализаций, обладающая поддержкой кастомных датасетов, обеспечивает хороший баланс между точностью и скоростью работы, что делает её подходящей для мобильных и встраиваемых систем.
- YOLOv12 – экспериментальная модель нового поколения, сочетающая сверточные и трансформерные элементы. Она включает гибридный backbone с attention-механизмами, обеспечивая более глубокое понимание контекста изображения.
- RF-DETR (Relation Former DETR) – это трансформерная модель, вдохновлённая архитектурой DETR, но улучшенная за счёт внедрения relation-aware декодеров. В отличие от YOLO, RF-DETR полностью отказался от anchor-боxов и использует механизм соответствия объектов (Hungarian Matching). Это позволяет достигать высокой точности без необходимости в постобработке типа Non-Maximum Suppression (NMS).

В отличие от традиционных архитектур, таких как Faster R-CNN, которые требуют более сложной настройки и зачастую не обеспечивают реального времени, YOLOv8 и YOLOv12 способны демонстрировать конкурентоспособную точность даже на относительно небольших и специализированных датасетах. Выбор данных моделей обусловлен их способностью эффективно работать с ограниченным количеством классов, высокой скоростью инференса и адаптируемостью к прикладным задачам транспортной инфраструктуры.

Основные компоненты архитектуры YOLOv8:

Backbone (CSPDarknet / EfficientNet-like): YOLOv8 использует усовершенствованную версию CSPDarknet с элементами эффективных сверточных блоков, заимствованных из EfficientNet. Это обеспечивает

высокую скорость обработки при сохранении качества извлечения признаков.

Neck (BiFPN): Для агрегации признаков с разных уровней применяется BiFPN (Bidirectional Feature Pyramid Network), обеспечивающий усиленную передачу информации между слоями и улучшение детекции объектов различных масштабов.

Head (Anchor-free Detection Head): В YOLOv8 используется anchor-free подход к предсказаниям, что упрощает архитектуру и повышает гибкость при локализации объектов. Модель напрямую предсказывает координаты и классы объектов.

Метод обучения: YOLOv8 применяет автоматическую настройку learning rate, а также продвинутые методы аугментации: Mosaic, MixUp, HSV-изменения, случайные обрезки и перевороты.

Оптимизация: Оптимизирована под PyTorch, может быть экспортирована в ONNX, TensorRT и другие форматы, что делает модель пригодной для встраиваемых решений и реального времени.

Основные компоненты архитектуры YOLOv12[3]:

Backbone (ConvFormer Hybrid): YOLOv12 использует гибридную архитектуру, сочетающую сверточные слои с трансформерными блоками. Это позволяет обрабатывать как локальные, так и глобальные признаки изображения.

Neck (Dynamic Path Aggregation): Усовершенствованный механизм передачи признаков основан на динамической агрегации, адаптирующейся к масштабу и плотности объектов.

Head (Adaptive Detection Head): Используется адаптивный head с интеграцией attention-механизмов, повышающих точность классификации и локализации.

Метод обучения: Модель обучается с использованием оптимизаторов нового поколения (например, Lion), механизма динамического балансирования потерь и обучающих стратегий на основе curriculum learning.

Оптимизация: YOLOv12 ориентирован на производительность в реальном времени на GPU, а также обладает потенциальной поддержкой TPU и FPGA.

Основные компоненты архитектуры RF-DETR:

Backbone (ResNet / ConvNext / Swin): Модель RF-DETR допускает использование различных backbones, включая ResNet или Swin-Transformer, для извлечения признаков из входного изображения.

Neck (Absence): В классической архитектуре RF-DETR отсутствует выделенный neck: признаки напрямую подаются в трансформер, что снижает сложность и количество гиперпараметров.

Head (Transformer Decoder + FFN): Основу head составляет декодер на базе многоголового внимания. Каждая позиция предсказания соответствует отдельному learnable query. FFN (Feed-Forward Network) обрабатывает выход трансформера для предсказания координат и классов объектов.

Метод обучения: Модель обучается end-to-end без необходимости в anchor-boxes или NMS. Используется Hungarian Matching Loss для сопоставления предсказаний и ground truth.

Оптимизация: Оптимизирована под обучение на больших датасетах. Поддерживает mixed precision training (FP16), хорошо масштабируется на многогPU.

IV. Сравнение

Эффективность обученных моделей оценивается на основе качества работы классификационной части и точности локализации, которая измеряется с помощью Intersection over Union (IoU) для каждой найденной детекции [4].

Вычисляются следующие метрики:

- $Precision = \frac{TP}{TP+FP}$ – характеризует долю истинно положительных срабатываний (правильно распознанных дорожных знаков) среди всех обнаруженных объектов,
- $Recall = \frac{TP}{TP+FN}$ – измеряет способность модели находить все целевые объекты на изображении,

где:

- TP (True Positive) – количество объектов, которые модель верно выявила и правильно классифицировала как целевые.
- FP (False Positive) – количество объектов, которые модель ошибочно приняла за целевые, хотя на самом деле ими не являются.
- FN (False Negative) – количество объектов, которые модель не обнаружила, несмотря на их наличие на изображении.

Для анализа эффективности обученных моделей YOLOv8, YOLOv12 и RF-DETR использовались также следующие метрики:

- Показатели mAP (mean Average Precision) предоставляют комплексную оценку модели, учитывая как точность обнаружения объектов, так и степень совпадения предсказанных и истинных областей с помощью индекса перекрытия (IOU).
    - mAP50 – это среднее значение точности при пороге IOU равном 0.5.
    - mAP50-95 – это среднее значение точности, рассчитанное при порогах IOU от 0.5 до 0.95 с шагом 0.05.
- Скорость обработки. При оценке производительности моделей в режиме

реального времени учитывались следующие показатели:

- Preprocess – время, затрачиваемое на предобработку изображения.
- Inference – время, необходимое для выполнения детекции.
- Postprocess – время обработки результатов детекции.

Высокая скорость обработки является ключевым фактором для систем реального времени, таких как ADAS и автономные транспортные средства.

Метрики обеспечивают комплексную оценку качества функционирования модели, учитывая как точность и полноту детекции, так и вычислительную эффективность. Применение нескольких метрик позволяет проводить объективное сравнение моделей и выбирать наиболее подходящую для решения конкретной задачи.

Оценка производительности моделей осуществлялась с использованием стандартных метрик, таких как Precision, Recall, mAP50, mAP50-95, а также с учетом вычислительных затрат, включающих время предобработки, инференса и постобработки.

Модель YOLOv8 показала следующие результаты:

- Precision: 0.95778
- Recall: 0.92608
- mAP50: 0.96761
- mAP50-95: 0.76975

Среднее время обработки одного изображения:

- Предобработка: 2.05 мс
- Инференс: 10.5 мс
- Постобработка: 6.13 мс

Модель YOLOv8 продемонстрировала высокий уровень точности и полноты распознавания при умеренном времени инференса. Такая производительность делает YOLOv8 привлекательной для задач, требующих высокой скорости при сохранении точности.

Результаты YOLOv12:

- Precision: 0.972
- Recall: 0.96343
- mAP50: 0.96697
- mAP50-95: 0.76478

Среднее время обработки одного изображения:

- Предобработка: 0.8 мс
- Инференс: 15.9 мс
- Постобработка: 0.7 мс

Модель YOLOv12 улучшила показатели точности, что свидетельствует о более надежном обнаружении целевых объектов. Значения mAP50 и mAP50-95 сопоставимы с результатами YOLOv8. Однако время инференса увеличилось, несмотря на уменьшение времени пред и постобработки.

Метрики модели RF-DETR:

- Precision: 0.9736
- Recall: 0.94532
- mAP50: 0.94341
- mAP50-95: 0.81531

Среднее время обработки одного изображения:

- Предобработка: 2.3 мс
- Инференс: 35.4 мс
- Постобработка: 4.17 мс

Модель RF-DETR показала лучший баланс между метриками качества и mAP50-95, что превосходит результаты обеих моделей YOLO. Precision и Recall составили 0.9736 и 0.94532 соответственно, что подтверждает высокую эффективность модели в задачах детекции. Однако время инференса оказалось значительно больше – 35.4 мс, что может ограничивать использование модели в системах реального времени.

Таблица 1 отображает количественные оценки для двух подходов.

ТАБЛИЦА 1. Оценка детектирующей части

Метрика	**YOLOv8**	**YOLOv12**	**RF-DETR**
Precision	0.95778	0.972	0.9736
Recall	0.92608	0.96343	0.94532
mAP50	0.96761	0.96697	0.94341
mAP50-95	0.76975	0.76478	0.81531
Время предобработки (мс)	2.05	0.8	2.3
Время инференса (мс)	10.5	15.9	35.4
Время постобработки (мс)	6.13	0.7	4.17

YOLOv12 демонстрирует улучшение ключевых метрик качества по сравнению с YOLOv8, что делает ее предпочтительным выбором для приложений, где приоритетом является максимальная точность и полнота обнаружения объектов, например, в системах автономного вождения и ADAS. В то же время YOLOv8 сохраняет преимущество в скорости обработки за счёт более быстрого инференса, что может быть критично для задач с жёсткими требованиями к времени отклика. Модель RF-DETR выделяется на фоне обеих моделей более высоким значением mAP50-95, свидетельствующим о лучшем качестве локализации и распознавания даже при более строгих порогах IoU. Однако более высокое время инференса RF-DETR ограничивает её применение в системах реального времени.

V. Заключение

В данной статье проведён комплексный сравнительный анализ трёх современных моделей детекции объектов – YOLOv8, YOLOv12 и RF-DETR – применительно к задаче распознавания дорожных знаков, связанных с железнодорожными переездами. Тестирование на уникальном датасете, содержащем три класса знаков, показали, что все рассмотренные модели способны эффективно решать поставленную задачу с высоким уровнем точности.

- YOLOv8 превосходит YOLOv12 по метрикам точности и скорости обработки (mAP50 0.9676 против 0.9670, среднее время инференса 10.5 мс против 15.9 мс), что делает её оптимальным выбором для приложений с жёсткими требованиями к быстродействию при сохранении высокого качества детекции. Высокие показатели Precision и Recall подтверждают её надёжность в задачах распознавания дорожных знаков.
- YOLOv12 демонстрирует улучшенную точность по сравнению с YOLOv8 (Precision 0.972 против 0.9578, Recall 0.9634 против 0.9260), что делает её предпочтительным вариантом в случаях, когда критична максимальная полнота обнаружения и качество локализации объектов. Несмотря на более высокое время инференса, она подходит для систем с менее жёсткими требованиями по времени отклика, где важнее точность.
- RF-DETR отличается самой высокой способностью к обобщению и точности локализации (mAP50-95 0.8153 против 0.7697 у YOLOv8 и 0.7648 у YOLOv12), что обеспечивает лучший баланс между качеством распознавания и устойчивостью к сложным условиям. Однако увеличенное время инференса (35.4 мс) ограничивает её использование в системах с необходимостью мгновенного отклика, хотя она отлично подходит для приложений, где приоритетом является точность и комплексный анализ изображений.

Полученные результаты подтверждают потенциал современных нейросетевых архитектур для интеграции в интеллектуальные транспортные системы и системы помощи водителю, обеспечивая повышение безопасности и эффективности движения вблизи железнодорожных переездов.

Литература

[1] Redmon J. Unified, real-time object detection //Proceedings of the IEEE conference on computer vision and pattern recognition. – 2016.

[2] Использование 3D-сетей для «предсказания» моделей поведения транспортных средств в задаче беспилотного движения трамвая / Н. С. Гужва, В. Е. Прун, В. В. Постников [и др.] // XXIX Санкт-Петербургская международная конференция по интегрированным навигационным системам : сборник материалов, Санкт-Петербург, 30 мая – 01 2022 года. – Санкт-Петербург: "Концерн "Центральный научно-исследовательский институт "Электроприбор", 2022. – С. 304-310. – EDN JQNIIU.

[3] Al Rabbani Alif, Mujadded & Hussain, Muhammad. (2025). YOLOv12: A Breakdown of the Key Architectural Features. 10.48550/arXiv.2502.14740.

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

ГРНТИ 20.53.19, 28.23.37
ASJC 1707
OECD 1.02 EP

Нейросетевые методы для распознавания дефектов в дорожном покрытии

И. Б. Бахвалов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2002622@edu.misis.ru

А. А. Кузьменко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2009361@edu.misis.ru

Аннотация — В данной работе рассматривается задача детекции и классификации дорожных дефектов, таких как линейные и мозаичные трещины дорожного покрытия. Задача имеет важное значение для обеспечения безопасности движения по дорогам и контроля за их состоянием. Для решения задачи применялись современные SOTA-модели – YOLO12L и RT-DETR Large, RF-DETR. Для обучения и тестирования был собран и размечен датасет, составленный из изображений, взятых из открытых источников и отражающих различные условия съемки. Результатом статьи является сравнение метрик качества дообученых моделей для решения задачи распознавания и классификации воздушных объектов.

Ключевые слова — Компьютерное зрение, Детекция объектов, Классификация трещин, Распознавание дорожных трещин, YOLO, RT-DETR, RF-DETR.

I. Введение

В последние годы задачи распознавания и классификации объектов на дорогах приобрели особую актуальность в связи с ростом использования систем автоматизированного управления транспортом и интеллектуальных транспортных систем. В этом контексте методы компьютерного зрения выступают эффективным инструментом для обнаружения таких объектов [1]. Современные алгоритмы обработки изображений и машинного обучения позволяют системам анализировать визуальные данные — фотографии и видеопотоки — и распознавать транспортные средства по их характерным визуальным признакам, таким как форма, размер и особенности движения.

Технологии глубокого обучения и нейросетевые модели, такие как YOLO [2] и DETR [3], показывают значительные достижения в области компьютерного зрения и успешно используются для задач обнаружения и классификации объектов на изображениях и видео. YOLO (You Only Look Once) [5] отличается высокой скоростью обработки и эффективностью, что делает её предпочтительным выбором для приложений с ограниченными вычислительными ресурсами и требованиями к быстродействию [6]. В то же время модели на основе трансформеров, такие как RT-DETR [7] и его улучшенные версии, обеспечивают высокую точность и лучшее понимание контекста сцены благодаря механизму внимания, что особенно важно в сложных условиях наблюдения [8].

Для успешного применения этих моделей в задачах распознавания различных видов трещин необходимо учитывать множество факторов, таких как разнообразие условий освещения, влияние погоды, а также особенности форм и движений объектов. Кроме того, ограниченность и неоднородность существующих датасетов затрудняют обучение универсальных моделей, которые могли бы эффективно работать в реальных условиях. Поэтому важно создавать и использовать разнообразные размеченные наборы данных, которые отражают реальные условия воздушного пространства.

II. Наборы данных

Для решения задачи детектирования и классификации воздушных объектов был собран и размечен пользовательский датасет, состоящий из изображений, взятых из открытых источников. Датасет включает 1842 изображений, для каждого из которых имеется аннотация в виде ограничивающих многоугольников, точно локализирующих объекты и относящих их к одному из двух классов: linear-crack или mosaic-crack.

Для оптимальной оценки качества моделей данные были разделены на тренировочную, валидационную и тестовую выборки в пропорциях 70%, 20% и 10% соответственно, что соответствует современным стандартам машинного обучения и позволяет эффективно контролировать процесс обучения и обобщающую способность моделей.

Каждое изображение датасета имеет расширение 400×400 пикселей, что обеспечивает однородность входных данных и упрощает этапы предобработки. Размеры объектов на изображениях варьируются от мелких, находящихся на заднем плане, до крупных, расположенных вблизи камеры. Такое разнообразие масштабов и дистанций повышает устойчивость модели к различным условиям съемки.

Для увеличения количества изображений в датасете была применена аугментация. Были применены фильтры поворота в разные стороны исходных изображений.

На рисунках 1–4 изображены примеры рассматриваемых данных в реальных условиях.

Рис. 1. Пример изображения из класса linear-crack

Рис. 2. Пример изображения с объектом из класса linear-crack

Рис. 3. Пример изображения с объектом из класса linear-crack

Рис. 4. Пример кадра с объектом класса mosaic-crack

III. Нейросетевые архитектуры

A. *YOLOv12 (You Only Look Once).*

YOLOv12 — это последняя версия популярной нейросетевой архитектуры для детекции объектов в реальном времени, которая продолжает развитие оригинальной модели YOLO. Эта модель сочетает в себе улучшенную точность и увеличенную скорость обнаружения, при этом сохраняя ключевые преимущества предыдущих версий – высокую производительность и универсальность применения. YOLOv12 разработана как единый фреймворк, поддерживающий множество задач компьютерного зрения, включая обнаружение, сегментацию и классификацию объектов, что делает её удобной и эффективной для широкого спектра приложений.

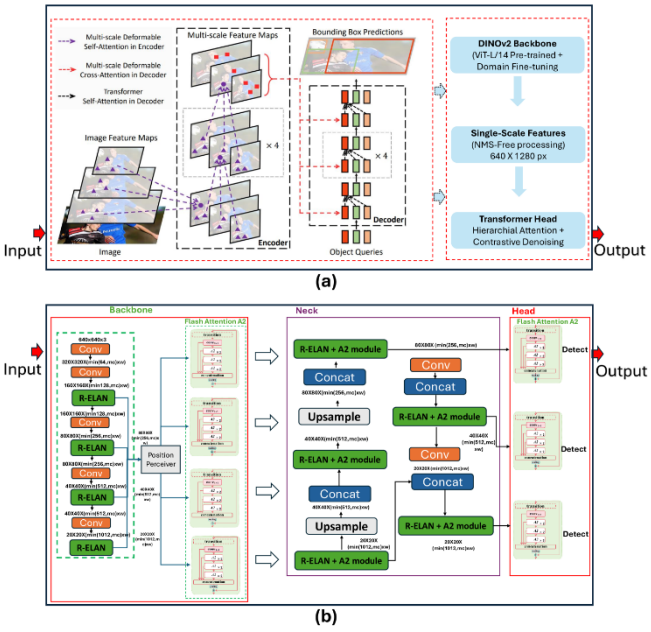


Рис. 5. Схематичное изображение архитектуры RF-DETR (a) и YOLOv12 (b)

Архитектура YOLOv12, как и её предшественники, построена по принципу одноступенчатого детектора, что означает одновременное выполнение задач локализации

и классификации объектов за один проход нейросети. В этой версии используется более глубокая и усовершенствованная сеть, что позволяет повысить точность распознавания при сохранении высокой скорости обработки.

Основные элемент архитектуры YOLOv12 включают:

- Backbone (основной блок):

  Основан на каскаде сверточных слоев с R-ELAN (улучшенная версия ELAN из YOLOv7). Обрабатывает изображение в 5 этапов с постепенным уменьшением разрешения. Каждый из этапов включает R-ELAN + A2 модуль, который комбинирует остаточные связи и механизм внимания для выделения ключевых признаков [9].

- Neck (средний слой):

  Динамический Attention-based Neck (аналог FPN/PAN, но с адаптивным вниманием): Upsample + Concat – многоуровневое слияние признаков; Flesh Attention A2 – перевзвешивает признаки, усиливая релевантные области (аналог BiFPN, но с вниманием) [10].

- Head (выходной слой):

  Detect-голова предсказывает bounding box'ы и классы. Использует NNS-free обработку (в отличии от классического NNS). Поддерживает иерархическое внимание для детекции объектов разного масштаба [11].

YOLOv12 включает в себя несколько усовершенствований, которые помогают улучшить точность и производительность модели. Одним из них является оптимизация вычислений. YOLOv12 использует более компактные и быстрые архитектуры для обработки изображений, что позволяет ускорить обучение и детекцию. Также модель использует многомасштабное обучение, т.е. она обучается с учётом объектов различных размеров, что улучшает её способность работать с малыми и крупными объектами на изображении.

B. RF-DETR (Refined Feature-aware DEtection TRansformers)

RF-DETR – это современный детектор объектов, основанный на архитектуре DETR, но с ключевыми усовершенствованиями, включая многоуровневое деформируемое внимание и контрастный денизинг, что позволяет достичь высокой точности детекции при сохранении эффективной работы. В отличие от классических DETR-моделей, RF-DETR использует предобученный DINOv2 (ViT-L/14) в качестве бэкбона, что обеспечивает мощное представление признаков, адаптированное под целевую задачу.

Основные элемент архитектуры RF-DETR включают:

- Backbone (основной блок):

  DINOv2 (ViT-L/14) - предобученный трансформер с дообучением под задачу. Обрабатывает изображение в одном масштабе (640×1280 px), но с деформируемым вниманием для анализа разных уровней детализации.

- Neck (средний слой):

  Многоуровневое деформируемое внимание в энкодере и декодере: Self-Attention (в энкодере) – анализирует признаки внутри изображения; Cross-Attention (в декодере) – связывает объектные запросы с картами признаков. Контрастный денизинг – фильтрация шумовых запросов для стабильного обучения.

- Head (выходной слой):

  Трансформер-голова с дискретными объектными запросами. Прямое предсказание bounding box'ов без явного NMS. Поддержка иерархического внимания для работы с работы с объектами разных размеров [11].

C. RT-DETR (Real-Time DEtection TRansformers)

RT-DETR – это современный детектор объектов в реальном времени, разработанный компанией Baidu и основанный на концепции DETR, но с рядом важных усовершенствований для повышения скорости и точности. В основе модели лежит гибридный кодировщик, который эффективно обрабатывает признаки изображения с разных масштабов, разделяя внутримасштабное взаимодействие признаков (AIFI) и межмасштабное слияние (CCFM). Это позволяет модели лучше учитывать разнообразие размеров объектов и снижать вычислительные затраты.

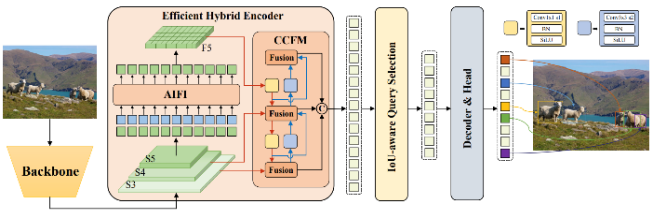


Рис. 6. Архитектура RT-DETR

Входными данными для кодировщика служат признаки с трёх последних этапов основной свёрточной сети (обычно уровни S3, S4, S5), что обеспечивает богатое представление изображения. Особенностью RT-DETR является использование механизма выбора запросов с учётом IoU (IoU-aware query selection), который помогает модели сосредоточиться на наиболее релевантных объектах, улучшая качество детекции.

Декодер модели итеративно оптимизирует объектные запросы, предсказывая ограничивающие рамки и оценки уверенности без необходимости в традиционной постобработке типа не максимального подавления (NMS). Такая архитектура позволяет RT-DETR работать эффективно и стабильно в режиме реального времени, обеспечивая гибкую настройку скорости вывода за счёт использования различных слоёв декодера без переобучения.

Кроме того, RT-DETR базируется на трансформерах зрения (Vision Transformer, ViT), что даёт модели возможность улавливать глобальный контекст изображения и улучшать качество детекции, особенно в

сложных сценах с множеством объектов и разнообразным фоном. Благодаря этому RT-DETR превосходит многие классические детекторы по точности локализации и устойчивости к вариативности данных, при этом сохраняя высокую производительность на ускоренных платформах, таких как CUDA с TensorRT.

Таким образом, RT-DETR сочетает в себе преимущества трансформерной архитектуры и эффективного гибридного кодировщика, обеспечивая высокую точность и скорость обнаружения объектов в реальном времени без сложных этапов постобработки.

IV. Сравнение

Для решения задачи детекции и классификации трещин на дорожном полотне были выбраны три модели из SOTA-подхода: YOLOv12, RF-DETR и RT-DETR. Обучение моделей проводилось в течение 60, 30 и 40 эпох соответственно. Для оптимизации использовался оптимизатор AdamW с начальными гиперпараметрами: learning rate (lr) - 0.01, weight_decay - 1e-4, что соответствует рекомендуемым значениям по умолчанию для соответствующих архитектур.

Для оценки качества работы моделей использовались метрики Precision, Recall и mAP@50. Метрика Precision отражает точность модели, показывая, какую долю из всех предсказанных объектов составляют действительно корректные обнаружения. Другими словами, она характеризует, насколько верны предсказания модели. Метрика Recall оценивает полноту модели, то есть способность находить все объекты интересующего класса на изображениях, показывая, какую часть от общего числа реальных объектов модель смогла обнаружить. Метрика mAP@50 (mean Average Precision при пороге IoU 0,5) объединяет информацию о точности и полноте, предоставляя комплексную оценку качества детекции объектов, учитывая степень совпадения предсказанных и истинных ограничивающих прямоугольников.

Для более наглядного понимания метрик введем следующие обозначения:

- TP (True Positive) – модель верно обнаружила объект нужного класса (дрон или птица).
- FP (False Positive) – модель ошибочно классифицировала объект другого класса или фон как целевой объект.
- FN (False Negative) – модель не обнаружила объект нужного класса, хотя он присутствовал на изображении [16,17].

Стоит отметить, что TN (True Negative) в задачах детекции обычно не применяется, так как она отражает количество правильно отвергнутых фонов, что не всегда релевантно для оценки качества детекции.

На основе этих величин рассчитываются основные метрики:

- $Precision = \frac{TP}{TP+FP}$ – показывает, какую долю всех предсказанных объектов модель определила корректно;
- $Recall = \frac{TP}{TP+FN}$ – отражает, какую часть всех присутствующих объектов модель смогла обнаружить.

В таблице 1 показаны количественные характеристики двух используемых подходов [12, 13]. Анализ представленных результатов показывает, что модель YOLOv12 превосходит RT-DETR???? по метрикам Precision и Recall, демонстрируя более высокую точность и полноту обнаружения объектов. В то же время RT-DETR достигает лучшего показателя по метрике mAP@50, что свидетельствует о более точной локализации объектов при заданном пороге IoU. Таким образом, YOLOv12 обеспечивает более сбалансированное обнаружение, тогда как RT-DETR выделяется высокой точностью определения границ объектов.

ТАБЛИЦА I. Оценка детектирующей части

	YOLOv12	**RT-DETR**	**RF-DETR**
Precision	87.2 %	83.7 %	91.1 %
Recall	74.8 %	74.1 %	70.0 %
mAP@50	82.2 %	83.4 %	81.2 %

На рисунке 7 показаны графики для модели YOLO, демонстрирующие изменение ключевых метрик обучения и валидации модели от числа эпох. На данных графиках видно, что присутствует стабильное снижение функций потерь как на тренировочных, так и на валидационных данных, что свидетельствует о корректном процессе обучения. Метрики качества же заметно быстро достигают высоких значений и остаются стабильными на протяжении большей части обучения, что указывает на эффективное выявление и классификацию объектов.

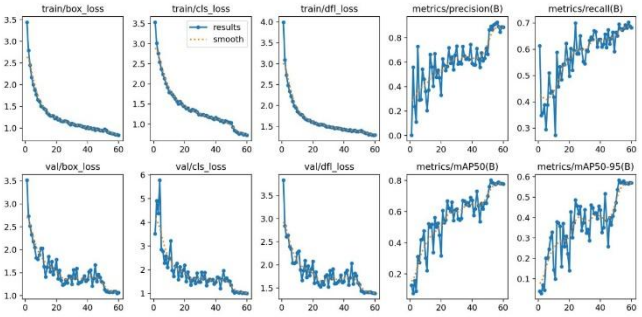


Рис. 7. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели YOLOv12

Аналогичный анализ был проведен для модели RT-DETR. Результаты представлены также в виде графиков на рисунке 8. На графиках, отражающих процесс обучения модели RT-DETR, также наблюдается устойчивое снижение всех функций потерь по мере увеличения числа эпох. В отличие от графиков обучения YOLOv12, на графиках для RT-DETR наблюдается более плавная динамика уменьшения функций потерь как на тренировочных, так и на валидационных данных. Это

говорит о более размеренном и устойчивом процессе обучения модели.

Кроме того, для RT-DETR характерно меньше выбросов и колебаний на графиках метрик качества, что свидетельствует о стабильности обучения и высокой устойчивости модели к внутренней вариативности датасета. Также такой характер графиков указывает на то, что модель RT-DETR менее подвержена переобучению или случайным ошибкам на отдельных эпохах.

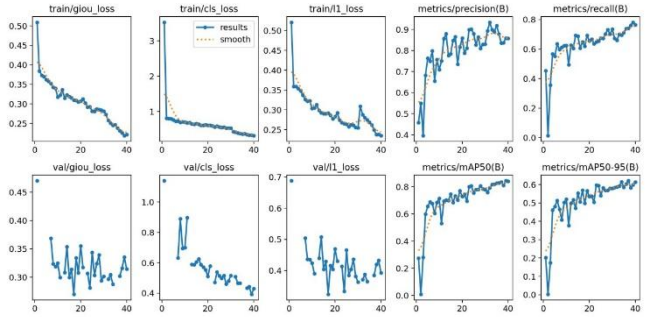


Рис. 8. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели RT-DETR

Подобный же анализ был проведен и для модели RF-DETR. Результаты представлены также в виде графиков на рисунке 9. На графиках, отражающих процесс обучения модели RF-DETR, наблюдается такое устойчивое снижение всех функций потерь по мере увеличения числа эпох. Однако, в отличие от графиков обучения YOLOv12, на графиках для RF-DETR наблюдается более плавная динамика уменьшения функций потерь как на тренировочных, так и на валидационных данных. Это говорит о более размеренном и устойчивом процессе обучения модели.

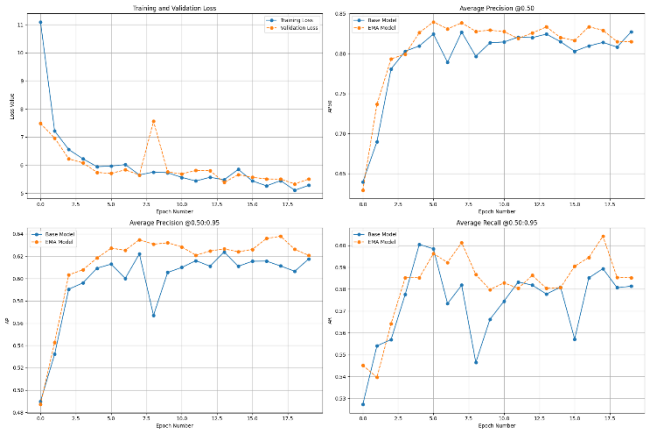


Рис. 9. Динамика изменения функции потерь и метрик качества на тренировочной и валидационной выборках при обучении модели RF-DETR

Таким образом, RT-DETR демонстрирует более сглаженное и стабильное обучение, что может быть преимуществом при работе с реальными, разнообразными данными в задачах, где важна предсказуемость поведения модели на новых примерах.

Экспериментальные результаты показали, что обе модели демонстрируют высокие показатели качества: YOLOv12 обеспечивает лучшие значения Precision (84,3%) и Recall (81,0%), что указывает на её высокую точность и полноту обнаружения объектов. RT-DETR, в свою очередь, достигает более высокого значения mAP@50 (88,6%), что свидетельствует о её превосходстве в точной локализации объектов. Графики обучения показывают, что YOLOv12 быстрее достигает высоких значений метрик, а RT-DETR отличается более плавным и стабильным процессом обучения с меньшим количеством выбросов.

Таким образом, выбор оптимальной модели зависит от конкретных требований задачи. YOLOv12 демонстрирует лучшие результаты по точности и полноте обнаружения, что делает её особенно привлекательной для задач, где важна оперативность и сбалансированное качество классификации. В то же время RT-DETR показывает превосходство в точной локализации объектов и отличается стабильностью обучения, что может быть критически важно при работе с разнообразными и сложными данными.

Результаты работы дообученых нейронных сетей YOLO, RT-DETR и RF-DETR представлены на рисунках 10-12,14-16 и 18–20. На рисунках 13, 17 и 21 представлены матрицы ошибок для YOLOv12, RT-DETR и RF-DETR.

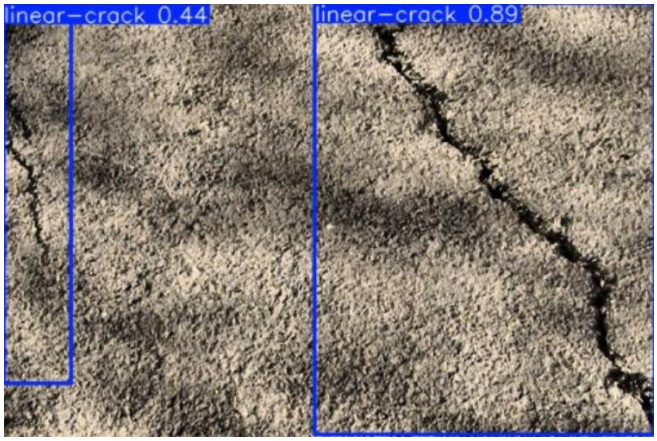


Рис. 10. Результат работы дообученной нейронной сети YOLOv12

Рис. 11. Результат работы дообученной нейронной сети YOLOv12

Рис. 12. Результат работы дообученной нейронной сети YOLOv12

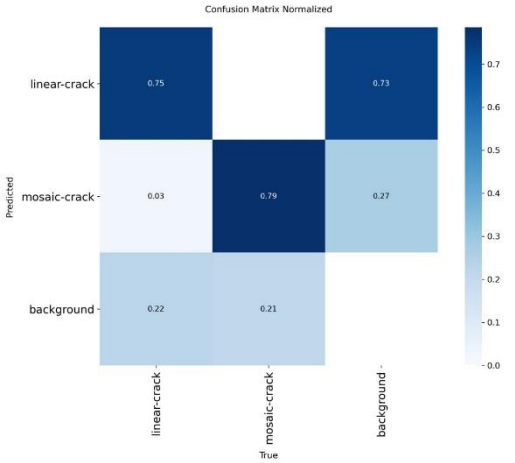


Рис. 13 Матрица ошибок YOLOv12

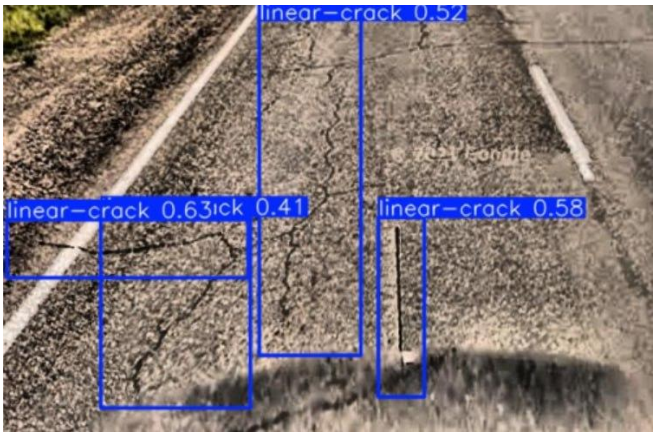


Рис. 14. Результат работы дообученной нейронной сети RT-DETR

Рис. 15. Результат работы дообученной нейронной сети RT-DETR

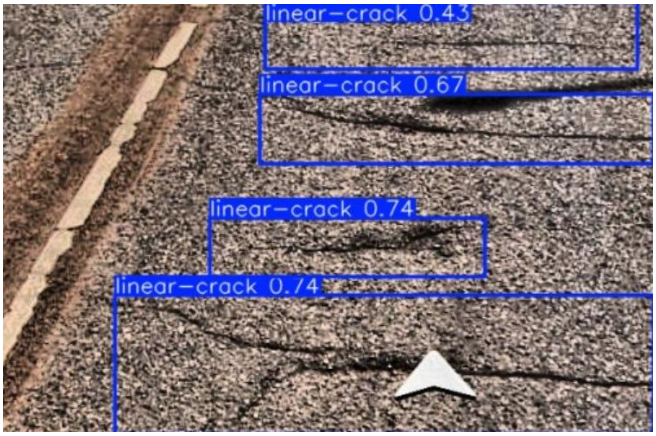


Рис. 16. Результат работы дообученной нейронной сети RT-DETR

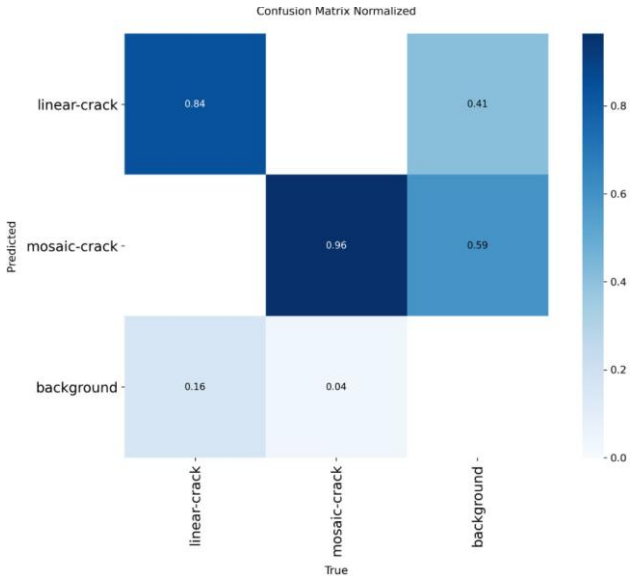


Рис. 17. Матрица ошибок нейронной сети RT-DETR

Рис. 18. Результат работы дообученной нейронной сети RF-DETR

Рис. 19. Результат работы дообученной нейронной сети RF-DETR

Рис. 20. Результат работы дообученной нейронной сети RF-DETR

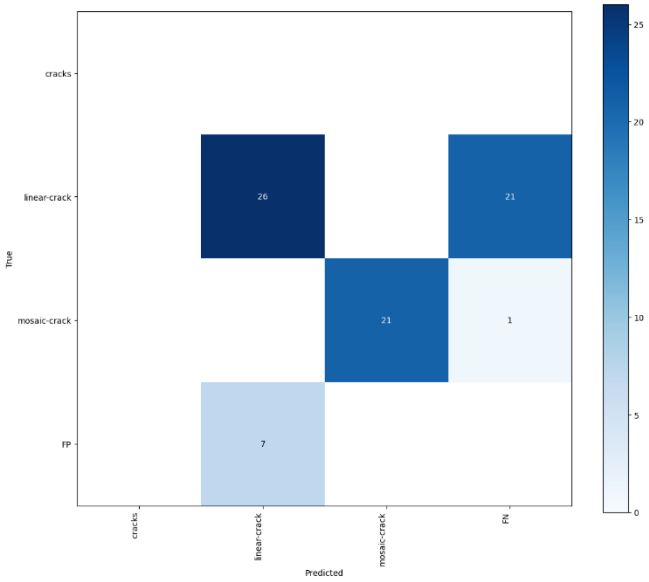


Рис. 21. Результат работы дообученной нейронной сети RF-DETR

V. Заключение

В данной работе были проанализированы три архитектуры нейронных сетей из числа современных SOTA-подходов — YOLOv12, RT-DETR и RF-DETR, применяемые для детекции и классификации дорожных трещин, как мозаичных, так и линейных. Для обучения и тестирования моделей был собран и размечен пользовательский датасет, состоящий из 1842 изображений с аннотациями для двух классов: «linear-crack» и «mosaic-crack». Оценка качества работы моделей проводилась с использованием стандартных метрик компьютерного зрения: Precision, Recall и mAP@50.

Результаты показали, что все три модели успешно справляются с задачей детекции и классификации дорожных дефектов. Модель YOLO немного уступает моделям DETR, так как может пропускать некоторые объекты на изображениях. В то же время модели DETR демонстрируют лучшую локализацию трещин. Таким образом, если приоритетом является высокая скорость и сбалансированное обнаружение объектов в реальном времени, то предпочтение стоит отдать модели YOLOv12. Если же задача требует максимальной точности локализации и устойчивости к изменениям в данных, более подходящими будут RT-DETR и RF-DETR. Полученные результаты могут быть полезны при выборе оптимальной архитектуры для практических задач мониторинга и обеспечения безопасности дорожного движения.

Литература

[1] Arya, H. Maeda, S. K. Ghosh, D. Toshniwal, A. Mraz, T. Kashiyama, and Y. Sekimoto, Deep learning-based road damage detection and classification for multiple countries, Automation in Construction, vol. 132, 2021.

[2] Ultralytics. YOLOv12: State-of-the-Art Object Detection Model [Electronic resource]. – GitHub repository. – 2023. – Available at: https://github.com/ultralytics/ultralytics (Accessed: 10 May 2025)

[3] Redmon J., Divvala S., Girshick R., Farhadi A. You Only Look Once: Unified, Real-Time Object Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2016. – P. 779–788.

[4] RT-DETR: Real-Time Detection Transformer [Electronic resource]. – GitHub repository. – 2024. – Available at: https://github.com/lyuwenyu/RT-DETR (Accessed: 10 May 2025)

[5] Ultralytics. YOLOv12: State-of-the-Art Object Detection Model [Electronic resource]. – GitHub repository. – 2023. – Available at: https://github.com/ultralytics/ultralytics (Accessed: 10 May 2025)

[6] Redmon J., Divvala S., Girshick R., Farhadi A. You Only Look Once: Unified, Real-Time Object Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2016. – P. 779–788.

[7] RT-DETR: Real-Time Detection Transformer [Electronic resource]. – GitHub repository. – 2024. – Available at: https://github.com/lyuwenyu/RT-DETR (Accessed: 10 May 2025)

[8] Carion N., Massa F., Synnaeve G., Usunier N., Kirillov A., Zagoruyko S. End-to-End Object Detection with Transformers // European Conference on Computer Vision (ECCV). – 2020. – P. 213–229.

[9] J Huang, K. Chen and Z. Liu, (2017). Speed/accuracy trade-offs for modern convolutional object detectors. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/1611.10012 (Accessed: 10 May 2025).

[10] J. Redmon, S. Divvala, R. Girshick, R and A. Farhadi, (2016). You Only Look Once: Unified, Real-Time Object Detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/15 06.02640(Accessed: 10 May 2025).

[11] J. Redmon and A. Farhadi, YOLO9000: Better, Faster, Stronger. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Available at: https://arxiv.org/abs/1612.08242 (Accessed: 10 May 2025).

[12] Towards Data Science, "Confusion Matrix and Performance Metrics," Towards Data Science Blog, https://towardsdatascience.com/an introduction-to-performance-metrics-in-machine-learning543bfa9256b1 (Accessed: Accessed: 10 May 2025

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

[14] zantsev A., Lepetit V., Fua P. Flying Objects Detection from a Single Moving Camera // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2017. – Vol. 39, No. 5. – P. 879–892.

[15] Kellenberger B., Marcos D., Tuia D. Detecting Mammals in UAV Images: Best Practices to Address a Substantially Imbalanced Dataset with Deep Learning // Remote Sensing of Environment. – 2018. – Vol. 216. – P. 139–153.

ГРНТИ 14.35.00
ASJC 1702
OECD 6.3.1

Сравнение моделей сегментации дорожных трещин на основе современных нейросетевых архитектур

А.О. Васильева
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2009903@edu.misis.ru

И. А. Ширеторова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2008125@edu.misis.ru

М. Гримм
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2007014@edu.misis.ru

***Аннотация*— автоматическое выявление трещин на дорожных покрытиях является важной задачей в области интеллектуального транспорта и инфраструктурного мониторинга. В данной работе рассматривается сравнение современных моделей сегментации — DeepLabV3+, BiSeNetV2 и YOLOv8-seg— с целью выявления наиболее эффективного подхода к обнаружению дорожных дефектов. Особенностью исследования является комплексная оценка моделей на широком спектре изображений с различными сложностями, что позволяет оценить их точность и устойчивость к визуальным искажениям, характерным для реальных дорожных сцен. Для обучения использован объединённый набор данных, составленный из нескольких открытых источников, таких как Crack500, CrackForest и датасет от Ultralytics, что обеспечивает высокое разнообразие изображений.**

Ключевые слова — сегментация изображений, дорожные дефекты, нейросетевые модели, DeepLabV3+, BiSeNetV2, YOLOv8, компьютерное зрение, детекция трещин, дорожная безопасность.

I. Введение

Состояние дорожной инфраструктуры оказывает прямое влияние на безопасность, комфорт и надёжность транспортной системы. Даже незначительные повреждения дорожного полотна, такие как микротрещины, при отсутствии своевременного обнаружения могут перерасти в серьёзные деформации покрытия, увеличивая риски аварий, затраты на обслуживание и сокращая срок службы дорожных конструкций [1,16]. Особенно остро эта проблема проявляется в условиях интенсивной эксплуатации, переменных климатических факторов и сезонных нагрузок, характерных для большинства регионов мира.

Традиционные методы диагностики дорожных покрытий, основанные на визуальном осмотре или специализированной технической аппаратуре, требуют значительных временных, трудовых и финансовых ресурсов, что затрудняет их масштабирование для регулярного мониторинга протяжённых дорожных сетей. В этой связи актуальным направлением становится разработка автоматизированных систем мониторинга с применением методов компьютерного зрения [2].

В последние годы особое внимание уделяется использованию глубинных нейросетевых архитектур для решения задач сегментации дорожных дефектов. Такие модели, как DeepLabV3+, основанная на атрибутивных свёртках и модулях пространственной пирамиды признаков, BiSeNetV2, совмещающая быстрое извлечение детальных и семантических признаков для работы в реальном времени, а также YOLOv8-seg — адаптированная модификация известной линейки YOLO для одновременной детекции и сегментации объектов, демонстрируют высокую эффективность в задачах выделения трещин различной сложности.

Дополнительно сложность задачи сегментации трещин обусловлена разнообразием их геометрических характеристик: ширина, длина, форма, разветвлённость, контрастность относительно фона покрытия. Трещины могут проявляться как изолированные тонкие разрывы, так и сложные переплетённые сети дефектов, что затрудняет их надёжную автоматическую идентификацию. На точность сегментации также существенно влияют внешние факторы: качество съёмки, угол обзора, освещение, тени, наличие загрязнений, шумов текстуры асфальта, горизонтальной разметки и инженерных швов.

В этом контексте современные нейросетевые архитектуры демонстрируют значительный прогресс благодаря способности обрабатывать изображения комплексно — извлекая как локальные, так и глобальные признаки структуры дорожного полотна. Глубокие сверточные сети позволяют учитывать пространственный контекст дефекта, а использование модулей пространственной агрегации и многоуровневых признаков увеличивает устойчивость моделей к шумам и неоднородностям изображения. За счёт этого становится возможным точное выделение даже микротрещин, что ранее было затруднено для традиционных алгоритмов обработки изображений.

II. Наборы данных

Для обучения и тестирования моделей сегментации в этом исследовании был сформирован составной набор данных, объединяющий изображения и соответствующие маски из трёх открытых источников: CrackForest, Crack500 и Ultralytics Crack Segmentation Dataset. Такой подход позволил достичь высокого разнообразия примеров дорожных трещин в различных условиях освещения, масштаба и текстуры покрытия. В общей сумме датасет насчитывает 2000 изображений. Для каждого изображения предоставлена соответствующая маска, где трещины

представлены в виде бинарного изображения (1 — трещина, 0 — фон). Маски представлены в виде изображений в оттенках серого и требуют минимальной предобработки для использования в моделях сегментации.

Данная сборка датасета уникальна тем, что имеет разные виды изображений:

- Типы трещин: продольные, поперечные, паукообразные (сетчатые), хаотично разбросанные мелкие трещины;
- Освещение: варьируется от яркого солнечного до пасмурного и теневого — что важно для моделирования реальных условий эксплуатации;
- Фон и шумы: присутствуют артефакты в виде листьев, дорожной разметки, тени от деревьев и других объектов, что делает задачу сегментации более приближённой к реальности;
- Текстура покрытия: варьируется от ровного асфальта до грубой и изношенной поверхности.

Датасет выложен в публичный доступ на Huggin Face.

Примеры изображений и масок к ним из сборочного датасета изображены на рисунке 1–2:

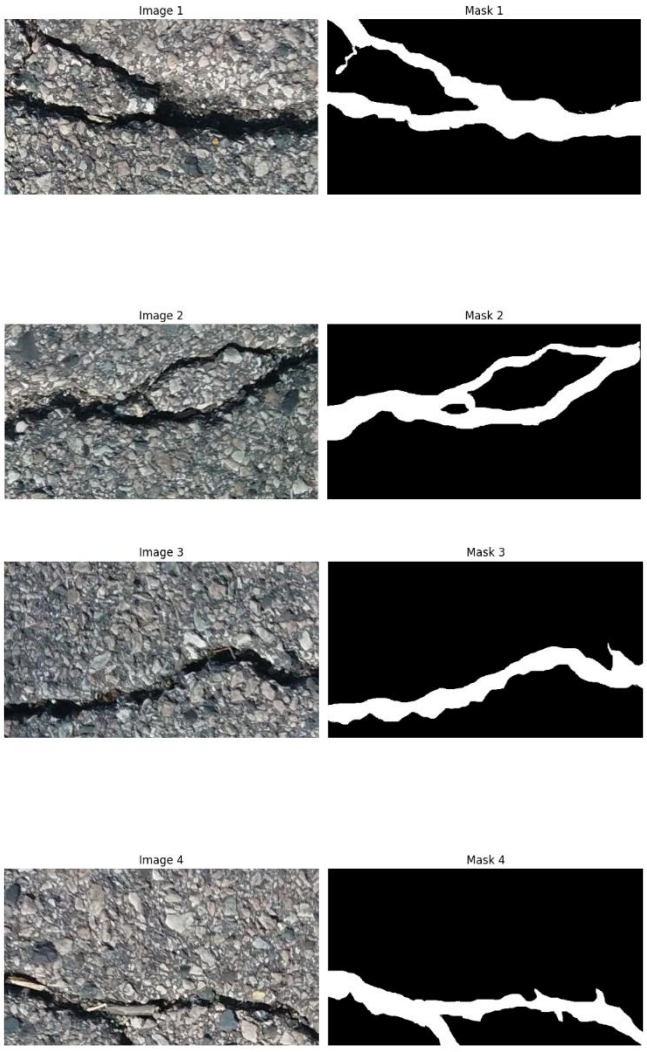


Рис. 1. Пример изображений датасета (1)

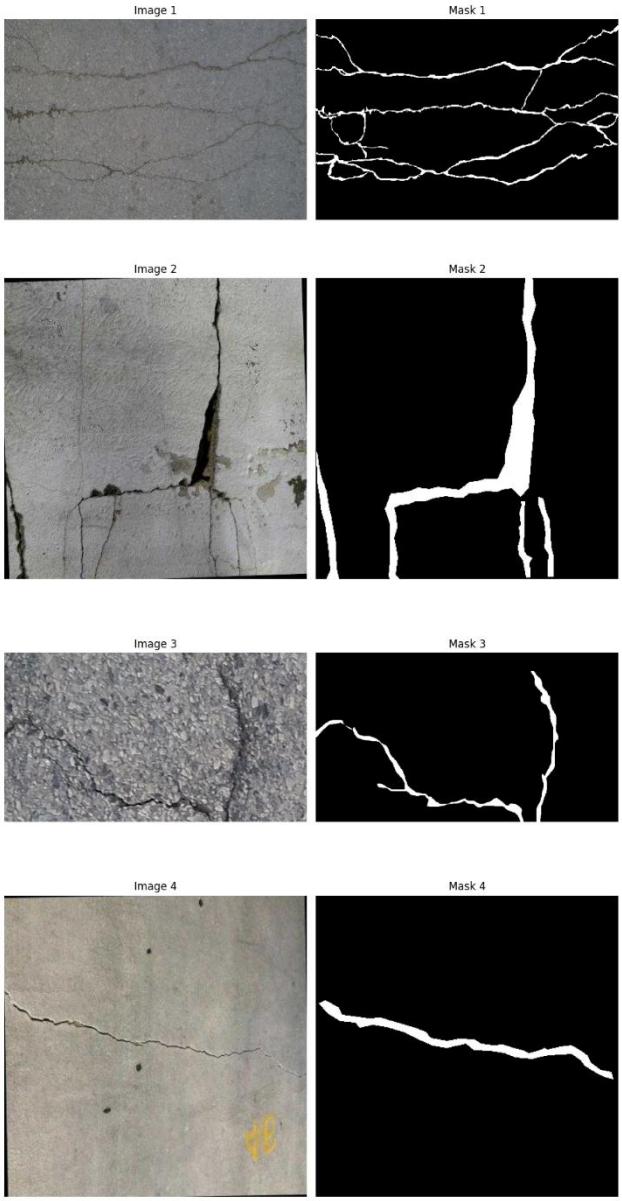


Рис. 2. Пример изображений датасета (2)

A. CrackForest

CrackForest является одним из классических наборов данных, предназначенных для задач бинарной сегментации дорожных трещин. Он содержит 118 изображений высокого разрешения. Снимки были сделаны на городских улицах Китая и опубликованы в 2016 году [3,17,19].

Данный датасет часто используется в научных работах как эталонная небольшая выборка для тестирования алгоритмов сегментации в условиях ограниченного объема данных. Он позволяет моделям работать как с тонкими одиночными трещинами, так и с разветвлёнными дефектами покрытия.

B. *Crack500*

Crack500 представляет собой более масштабный и разнообразный набор данных, содержащий 500 изображений. Изображения были сделаны у Главного кампуса Университета Темпл в США, а сам датасет был опубликован в 2019 году [4].

Особенностью Crack500 является более высокая сложность для алгоритмов: на части снимков присутствуют визуальные шумы в виде пятен масла, трещин неправильной формы и стыков покрытия, усложняющих процесс точной бинарной сегментации. Аннотации к датасету создавались вручную экспертами, что позволяет использовать набор как для обучения, так и для точного тестирования современных моделей глубокого обучения.

C. *Ultralytics Crack Segmentation Dataset*

Этот датасет создан на базе современных стандартов разметки и преимущественно ориентирован на городские и шоссейные дороги. Он состоит из изображений в формате 640×640 пикселей с уже готовыми бинарными масками. Для общей сборки было использовано ~ 1000 изображений. Преимущества этого датасета в том, что на многих изображениях присутствуют элементы дороги – обочины, бордюры, стыки плит, что позволяет моделям отличать реальные трещины от конструктивных линий. Аннотации производились с помощью полуавтоматических инструментов сегментации и ручной валидации. Датасет предоставлен компанией Ultralytics через платформу Roboflow был и опубликован в 2022 году [5,18].

III. Нейросетевые архитектуры

A. *DeepLabV3+*

DeepLabV3 - это усовершенствованная архитектура нейронной сети, разработанная для решения задачи семантической сегментации изображений. Этот метод предполагает присвоение каждому пикселю класса Версия DeepLabV3+ представляет собой усовершенствованную версию по сравнению со своими предшественниками из серии Deep Lab, обеспечивающее повышенную точность и эффективность сегментации сложных изображений. Эта последняя версия серии не только превзошла своего предшественника, DeepLabV3, но и достигла самых современных характеристик (SOTA) при пересечении среднего расстояния по сравнению с Union (mIOU) [6].

Encoder-Decoder:

- Encoder в DeepLabV3+ в первую очередь отвечает за извлечение семантической информации из изображения. Он использует модифицированную модель Xception, которая представляет собой мощную глубокую сверточную нейронную сеть, известную своей эффективностью и точностью. Кодировщик использует сложную свертку для увеличения поля зрения фильтров, позволяя охватить более широкий контекст без снижения пространственного разрешения карты объектов;
- Основная функция decoder заключается в уточнении результатов сегментации, особенно по границам объектов. Он извлекает грубые семантические признаки из кодера и постепенно уточняет их, комбинируя с низкоуровневыми признаками, полученными ранее в сети. Такое сочетание помогает запечатлеть мелкие детали и улучшает локализацию краев объекта.

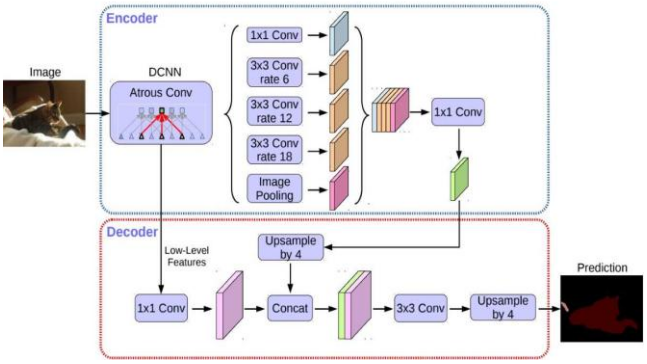


Рис. 3. Encoder-Decoder структура в DeepLabV3+

Основная особенность DeepLabV3+ заключается в применении технологии атрибутивных свёрток (Atrous Convolution), которые позволяют контролировать «область видимости» (receptive field) нейросети без увеличения числа параметров. Это особенно важно для задач, где необходимо обнаруживать как мелкие детали (например, тонкие трещины), так и крупные объекты [7].

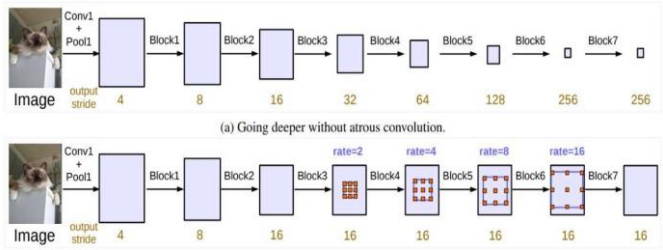


Рис.4. Два варианта построения сверточной нейросети: (а) без атрибутивной свёртки, (б) с атрибутивной свёрткой

На рисунке 4 сравниваются два варианта построения сверточной нейросети:

(a) Без атрибутивной свёртки:

- Сеть становится глубже, и на каждом уровне разрешение уменьшается (stride: 4 → 8 → 16 → 32 → 64...).
- Это приводит к потере пространственной точности: модель «видит» больше, но менее детально.
- Проблема особенно критична при сегментации мелких объектов, таких как трещины.

(b) С атрибутивной свёрткой (Atrous convolution):

- После блока 3 разрешение не уменьшается дальше (stride = 16 фиксируется).

- Вместо этого увеличивается параметр rate в свёртке:
  rate=2, rate=4, rate=8, rate=16
- Это означает, что свёртка «раздвигается» — ядро видит шире (больше контекста), но не теряет пространственную точность.

Получается, модель сохраняет высокое разрешение признаков и при этом способна анализировать глобальный контекст, что особенно полезно для сегментации структур, расположенных неравномерно

В рамках данного исследования модель DeepLabV3+ была обучена на собранном датасете дорожных трещин в течение 10 эпох при размере изображения 128×128 пикселей и размере батча 8.

B. YOLOv8-seg

Модель YOLOv8-seg представляет собой одну из модификаций популярной архитектуры YOLOv8, адаптированную для задач сегментации изображений. В отличие от своих предшественников, YOLOv8-seg совмещает в себе возможности как детектирования объектов, так и выделения их точных контуров с помощью масок [8, 14,15]. Такая гибридная структура особенно удобна в задачах instance segmentation, где требуется одновременно определить местоположение объекта и сегментировать его форму.

Архитектурно YOLOv8-seg расширяет стандартную модель YOLOv8 (рисунок 4) за счёт добавления отдельного сегментационного блока (Mask Head), при этом сохраняя общую структуру бэкбона и нейка. Основным отличием является использование специального декодера масок, что позволяет объединить задачи детекции объектов и сегментации их контуров в единую архитектуру.

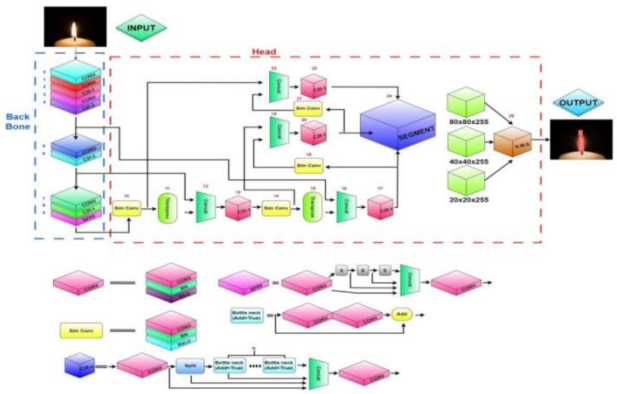


Рис. 5. Архитектура YOLOv8

В основе YOLOv8 лежит усовершенствованный CNN-бэкбон с использованием C2f-блоков (Cross-Stage Partial Connections), которые позволяют эффективнее передавать признаки между слоями, снижая потери информации на ранних этапах свёртки. Для сегментации дополнительно используется модуль mask head, который преобразует пространственные признаки из feature map в маски предсказания. Маски формируются на основе привязанных anchor-free предсказаний объектов и соответствующих им feature map с высокоразрешёнными признаками [9, 20].

Преимуществом YOLOv8-seg является высокая скорость обработки за счёт оптимизированной архитектуры, что делает модель удобной для применения в системах реального времени, например, в мобильных приложениях или системах контроля дорожного полотна с видеокамер. Однако особенностью архитектуры YOLOv8-seg остаётся привязка сегментации к объектной детекции, что при работе с тонкими, разветвлёнными структурами — такими как дорожные трещины — может ограничивать её точность по сравнению с специализированными сегментационными архитектурами (например, DeepLabV3+ или SegFormer) [10, 11].

В рамках данного исследования модель YOLOv8-seg обучалась на собранном датасете дорожных трещин в течение 50 эпох при размере изображения 128×128 пикселей и размере батча 8. Эти параметры позволили обеспечить баланс между скоростью обучения и стабильностью сходимости модели на доступных вычислительных ресурсах.

C. BiSeNetV2

Модель BiSeNetV2 (Bilateral Segmentation Network V2) представляет собой улучшенную версию быстрой архитектуры сегментации изображений, изначально разработанную для задач семантической сегментации в реальном времени. Основная идея архитектуры заключается в раздельной обработке пространственных и семантических признаков изображения с последующим их объединением для получения точных и детальных карт сегментации.

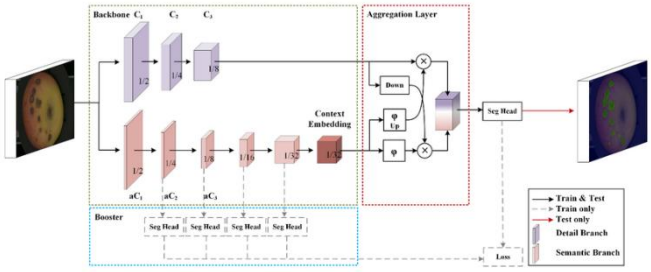


Рис. 6. Архитектура сети BiSeNetV2

BiSeNetV2 состоит из двух основных ветвей: Detail Branch и Semantic Branch. Detail Branch работает с высоким разрешением входных изображений и отвечает за сохранение мелких пространственных деталей, таких как границы объектов и тонкие элементы, что особенно важно при работе с трещинами дорожного полотна. Semantic Branch, напротив, обрабатывает изображение на пониженных разрешениях, извлекая глобальные семантические признаки сцены. После обработки обе ветви объединяются с помощью специального Bilateral Guided Aggregation модуля, который синхронизирует пространственную и контекстную информацию, обес-

печивая высокое качество сегментации при сохранении высокой скорости работы модели [12, 13].

Преимуществом BiSeNetV2 является оптимальный баланс между точностью и скоростью, что позволяет использовать данную архитектуру на мобильных устройствах, в системах автономного транспорта и в дорожных инспекционных комплексах. Модель демонстрирует высокую эффективность при сегментации объектов с чёткими границами и протяжёнными структурами, включая дорожные трещины различной формы.

Модель BiSeNetV2 была обучена на собранном датасете дорожных трещин в течение 10 эпох при размере изображения 256×256 пикселей и размере батча 8.

IV. Проведелние испытаний

Для оценки качества работы каждой из выбранных моделей сегментации — DeepLabV3+, BiSeNetV2 и YOLOv8-seg — было проведено тестирование на специально выделенной тестовой выборке из собранного датасета дорожных дефектов. В качестве тестового множества использовались изображения, ранее не задействованные при обучении моделей.

Для повышения достоверности результатов испытаний проводилось случайное выборочное тестирование. Из общего датасета случайным образом отбиралось 4 изображения, на которых для каждой модели выполнялось предсказание сегментационной маски. Предсказанные маски сравнивались с эталонной разметкой, по результатам чего рассчитывались ключевые метрики качества: Intersection over Union (IoU), точность (Precision), полнота (Recall) и F1-мера.

Кроме количественного анализа, дополнительно была выполнена визуальная оценка результатов сегментации на выбранных изображениях. Полученные сегментационные карты позволили проанализировать особенности работы каждой модели в условиях различных типов дефектов дорожного полотна, освещённости, текстур и помех.

ТАБЛИЦА 1. Оценка точности

	IoU	**Precision**	**Recall**	**F1-score**
DeepLabV3+	0.97	0.98	0.97	0.975
BiSeNetV2	0.75	0.77	0.83	0.85
YOLOv8-seg	0.65	0.68	0.63	0.65

На таблице 1 можем увидеть, что лучшие результаты продемонстрировала модель DeepLabV3+, достигнув IoU 0.97, точности 0.98 и полноты 0.97. Высокая точность обусловлена эффективной архитектурой с а́трибутивными свёртками и пространственно-пирамидальной агрегацией признаков, что позволяет точно выделять даже тонкие трещины.

Модель BiSeNetV2 показала сбалансированные результаты (IoU 0.85), успешно выделяя дефекты при высокой скорости обработки. Незначительное снижение полноты связано с пропуском некоторых узких или слабоконтрастных трещин.

YOLOv8-seg продемонстрировал наименьшие значения (IoU 0.65), что объясняется спецификой архитектуры, ориентированной в первую очередь на сегментацию полноразмерных объектов. Однако модель сохраняет высокую скорость и может использоваться для предварительного анализа.

Таким образом, DeepLabV3+ показал наилучшее SOTA-качество сегментации, тогда как BiSeNetV2 и YOLOv8-seg являются перспективными для систем, требующих высокой скорости обработки.

V. Сравнение

Рассмотрим получившиеся результаты на тестовом наборе нашего датасета.

На рисунке 7 представлено предсказание модели DeepLabV3+. Модель довольно точно воспроизводит контуры трещин, эффективно сегментируя как широкие, так и тонкие дефекты покрытия. Даже при наличии сложных фонов, теней или текстурных шумов DeepLabV3+ сохраняет высокую точность границ и полноту выделения трещин.

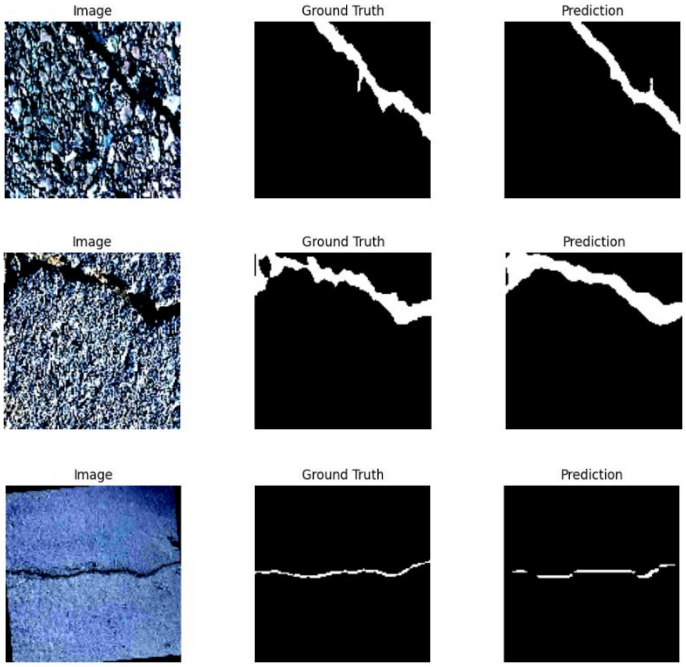


Рис. 7. Пример сегментации модели DeepLabV3+

YOLOv8-seg показывает менее точное воспроизведение сегментационных границ, иногда теряя фрагменты трещин или объединяя несколько линий в единый контур. Это объясняется тем, что изначально архитектура YOLO ориентирована преимущественно на детекцию объектов с чёткими границами и плохо адаптируется к сегментации протяжённых узких дефектов. В условиях сложного текстурного фона (например, на шершавом асфальте, при наличии стыков, бордюров, теней и следов шин) YOLOv8-seg периодически объединяет несколько разрозненных трещин в единый сегмент,

формируя чрезмерно широкие зоны выделения (рисунок 8).

Можно прийти к выводу, что для задач точной и детализированной сегментации сложных линейных структур трещин архитектура YOLO в текущей версии не совсем подходит.

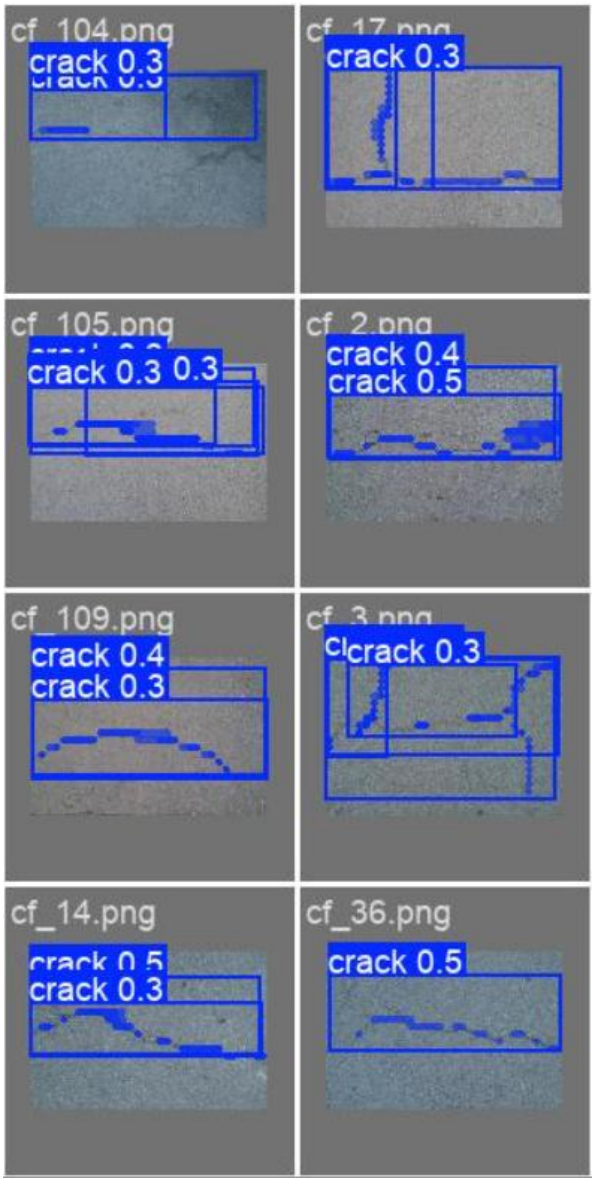


Рис. 8. Пример сегментации YOLOv8-seg

BiSeNetV2 демонстрирует сбалансированное качество сегментации при высокой вычислительной эффективности. Однако при работе с микротрещинами, которые характеризуются чрезвычайно тонкой геометрией и переменной контрастностью, модель иногда демонстрирует склонность к пропуску узких, слабовыраженных или частично размытых дефектов. Это обусловлено тем, что архитектура BiSeNetV2 делает акцент на скорость обработки, сокращая глубину извлекаемых признаков ради быстродействия, что ограничивает её способность восстанавливать сложные мелкие структуры трещин.

Тем не менее, учитывая высокую скорость инференса, компактность модели и небольшие требования к ресурсам, BiSeNetV2 остаётся перспективным кандидатом для применения в системах реального времени, особенно при первичном мониторинге дорожной сети, где ключевым фактором выступает скорость обработки потока данных.

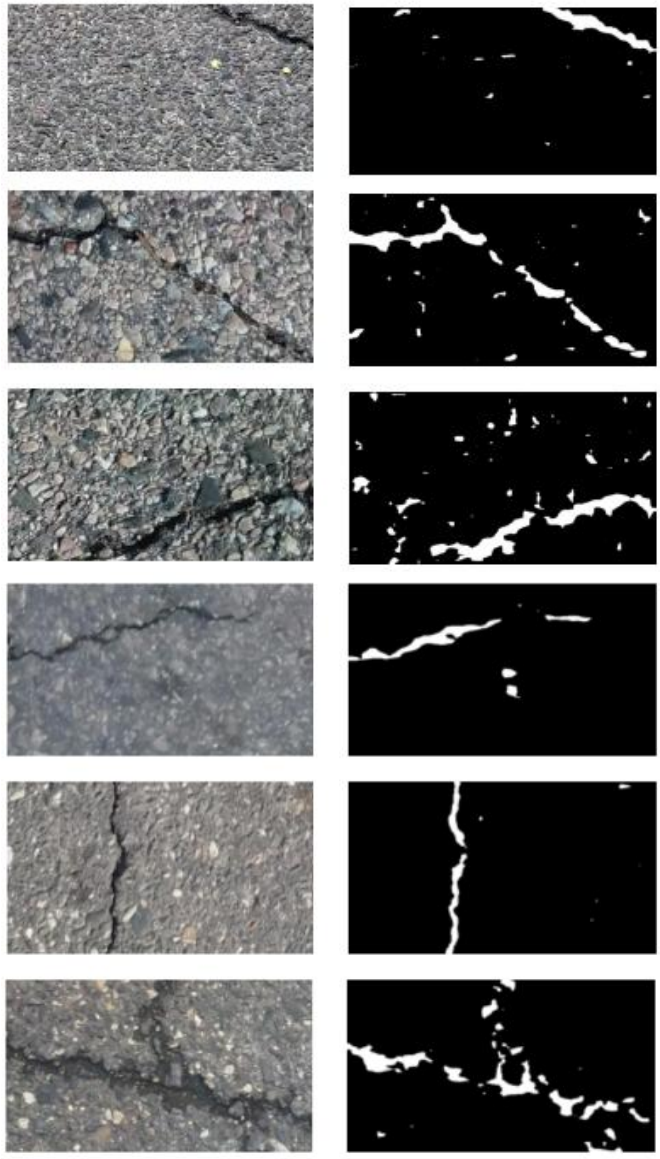


Рис. 9. Пример сегментации BiseNetV2

VI. Заключение

В данной работе проведено комплексное исследование методов сегментации дорожных дефектов с использованием современных нейросетевых архитектур DeepLabV3+, BiSeNetV2 и YOLOv8-seg. Для обучения моделей был сформирован объединённый датасет на основе нескольких открытых источников, что обеспечило разнообразие условий съёмки, освещения и типов дорожных покрытий.

Результаты численного и визуального анализа показали, что модель DeepLabV3+ демонстрирует наилуч-

шую точность сегментации микротрещин, обеспечивая state-of-the-art (SOTA) качество. Высокие значения IoU, точности и полноты подтверждают способность данной архитектуры к точному выделению как крупных, так и тонких дефектов дорожного полотна.

BiSeNetV2 показал сбалансированное сочетание качества сегментации и скорости инференса, что делает его перспективным решением для систем мониторинга в реальном времени, особенно в условиях ограниченных вычислительных ресурсов.

Модель YOLOv8-seg продемонстрировала высокую скорость обработки, но уступила по точности сегментации тонких линейных дефектов, что объясняется спецификой архитектуры, ориентированной на сегментацию замкнутых объектов с чёткими границами.

Полученные результаты подтверждают эффективность современных методов компьютерного зрения в задаче мониторинга состояния дорожного покрытия. Перспективными направлениями дальнейших исследований являются использование мультимодальных данных (лидар, ИК-камеры), обучение моделей на больших специализированных выборках с географической привязкой, а также внедрение адаптивных постобработок для повышения точности при анализе сложных дефектов

Литература

[1] A. A. Abakumov and V. O. Khuako, "Questions of road layer segmentation," in Artificial Intelligence in Industrial, Commercial, Medical, and Financial Applications: Proceedings of the Scientific and Technical Seminar of the Department of "Engineering Cybernetics", ser. 1. Moscow: National University of Science and Technology "MISIS", 2023, pp. 40. [Online]. Available: [http://sadekov.su/Articles/kik_2023.pdf]

[2] V. O. Kirvyakov "Questions of road layer segmentation" in Artificial Intelligence in Industrial, Commercial, Medical, and Financial Applications: Proceedings of the Scientific and Technical Seminar of the Department of "Engineering Cybernetics", ser. 1. Moscow: National University of Science and Technology "MISIS", 2023, pp. 146. [Online]. Available: [http://sadekov.su/Articles/kik_2023.pdf]

[3] Y. Shi, L. Cui, Z. Qi, F. Meng and Z. Chen, "Automatic Road Crack Detection Using Random Structured Forests," in *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 12, pp. 3434-3445, Dec. 2016, doi: 10.1109/TITS.2016.2552248.

[4] Cui, L., Qi, Z., Chen, Z., Meng, F., Shi, Y. (2015). Pavement Distress Detection Using Random Decision Forests. In: Zhang, C., *et al*. Data Science. ICDS 2015. Lecture Notes in Computer Science(), vol 9208. Springer, Cham.

[5] Yang F. et al. “CRACK500: A large-scale pavement crack dataset and benchmark”, Dataset description, 2019.

[6] Zhang M., Xu J. “A semantic segmentation model for road cracks combining channel–space convolution and frequency feature aggregation”, Scientific Reports, 2024, vol. 14, article 16038

[7] Zhang Z., He Y., Hu D. et al. “Algorithm for pixel-level concrete pavement crack segmentation based on an improved U-Net model”, Scientific Reports, 2025, vol. 15, article 6553.

[8] Zhang Y., Liu C. “MixCrackNet: Network for robust and high-accuracy pavement crack segmentation”, Automation in Construction, 2024, vol. 162, article 105375.

[9] Zhang C., Bahrami M., Mishra D.K., Yuen M.M.F. “SelectSeg: Uncertainty-based selective training and prediction for accurate crack segmentation under limited data and noisy annotations”, Reliability Engineering & System Safety, 2025, vol. 259, article 110909.

[10] Ali L., AlJassmi H., Swavaf M. et al. “RS-Net: Residual Sharp U-Net architecture for pavement crack segmentation and severity assessment”, Journal of Big Data, 2024, vol. 11, article 116.

[11] Zou Q., Zhang Z., Li Q. et al. “DeepCrack: Learning hierarchical convolutional features for crack detection”, IEEE Transactions on Image Processing, 2019, vol. 28, no. 3, pp. 1498–1512.

[12] Yang F., Zhang L., Yu S. et al. “Feature Pyramid and Hierarchical Boosting Network for pavement crack detection”, IEEE Transactions on Intelligent Transportation Systems, 2019, vol. 20, no. 11, pp. 3846–3859.

[13] Wang P., Zhu J., Zhu M. et al. “Fast and accurate semantic segmentation of road crack video in a complex dynamic environment”, International Journal of Pavement Engineering, 2023, vol. 24, no. 1, pp. 1–16.

[14] Asadi Shamsabadi E., Erfani S.M.H., Xu C., Dias-da-Costa D. “Efficient semi-supervised surface crack segmentation with small datasets”, Automation in Construction, 2024, vol. 154, article 105015.

[15] Hu Y., Yang X., Li K., Xiao S. “Deep learning-based intelligent detection of pavement distress: A bibliometric review”, Advanced Engineering Informatics, 2024, vol. 60, article 101785.

[16] Liu J., Chen R., Yan H. “Real-time pavement crack detection with a lightweight CNN”, Measurement, 2024, vol. 219, article 113459.

[17] Chen W., Gao X., Rong Y. “Lightweight high-accuracy model for crack segmentation on edge devices”, Applied Sciences, 2022, vol. 14, no. 24, article 11632.

[18] Zheng T., Zhou L., Wang J. “YOLOv8-ES: Efficient and accurate road crack damage detection”, Journal of Intelligent & Robotic Systems, 2025 (in press).

[19] Sun B., Liu C., Zhang P. “System design of vehicle-mounted road crack identification based on deep learning”, Proceedings of the International Conference on Machine Vision 2024, pp. 254–259.

[20] Rathnakumar R., Liu Y. “Deep learning in crack detection: A comprehensive scientometric study”, Engineering Reports, 2025.

ГРНТИ 41.51.41
ASJC 1707
OECD 01.02.EP

Исследование возможности распознавания и классификации галактик на астрономических снимках с помощью методов компьютерного зрения

П. И. Дорошев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m24115158@edu.misis.ru

М.А. Хижняк
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2414908@edu.misis.ru

***Аннотация*— В данной работе исследуется возможность применения современных архитектур глубокого обучения, таких как DETR (Detection Transformer) и YOLO (You Only Look Once), а также RT-DETR, для автоматического обнаружения и классификации галактик на астрономических снимках. Большое внимание уделено созданию обучающего набора данных на основе изображений из проекта Sloan Digital Sky Survey (SDSS) и их разметке с использованием классификации галактик из проекта GalaxyZoo. Проведено обучение моделей DETR, YOLO и RT-DETR, оценена их эффективность в задачах детектирования и классификации галактик. Результаты работы демонстрируют перспективность использования трансформерных архитектур и сверточных нейронных сетей для автоматизации анализа астрономических изображений.**

Ключевые слова — Компьютерное зрение, Сверточные нейронные сети, Трансформеры, Детектирование галактик, Астрономия.

I. Введение

В условиях стремительного развития технологий искусственного интеллекта в настоящее время особое внимание уделяется исследованиям в области компьютерного зрения, которые охватывают широкий круг предметных областей. Таким областями, например, могут быть транспортные системы [1] [2], экология [3] [4], зоология [5] и многие другие. Также особый интерес представляет исследование возможности использования методов компьютерного зрения в области астрономии.

Современные астрономические проекты, такие как Hubble Space Telescope (HST), James Webb Space Telescope (JWST) или Large Synoptic Survey Telescope (LSST), генерируют огромные объемы данных, содержащих изображения миллионов галактик, звезд и других космических объектов. Например, проект Euclid, запущенный Европейским космическим агентством и направленный на изучение темной материи и темной энергии, будет генерировать около 100 гигабайтов сжатых данных в день [6].

Обработка таких данных вручную становится все более сложной задачей, что стимулирует развитие методов компьютерного зрения и глубокого обучения для автоматизации анализа астрономических изображений. Одной из ключевых проблем является распознавание морфологических типов галактик, таких как эллиптические, спиральные или взаимодействующие системы. Традиционные подходы, основанные на визуальном анализе, требуют значительных временных затрат и могут страдать от субъективности.

Современные подходы, основанные на глубоком обучении, демонстрируют большие перспективы в области автоматического анализа астрономических изображений. В частности, сверточные нейронные сети (CNN) и показали хорошие результаты в задачах обнаружения следов астероидов [7], гравитационных линз [8] и космических объектов в целом [9].

Также в последнее время популярны детектирующие модели, использующие архитектуру трансформера, изначально созданную для задач обработки естественного языка. Такие модели демонстрируют хорошие показатели точности в различных сферах применения, однако работ, исследующих возможность применения такой архитектуры в задаче обработки астрономических изображений, немного. Например, в работе [10], детектирующий трансформер DETR применяется для задачи обнаружения околоземных объектов.

Так, в данной работе исследуется возможность применения современных архитектур для автоматического обнаружения и классификации галактик на астрономических снимках. Также большое внимание уделяется процедуре создания и доработки обучающего набора данных на основе астрономических снимков, полученных из репозитория проекта SDSS. Дополнительно рассматриваются различные технологии компьютерного зрения с целью выбора оптимального решения, способного предоставить необходимую точность при высокой эффективности.

Основными задачами работы являются:

1. Создание набора данных на основе астрономических снимков, включающего размеченные изображения галактик в формате, пригодном для обучения нейросетевых моделей.

2. Трансформация датасета в форматы, требуемые различными архитектурами и переобученными моделями.
3. Обучение моделей на основе архитектур DETR и YOLO для задачи детектирования и классификации галактик.
4. Оценка эффективности предложенных методов.

II. Набор данных

A. Sloan Digital Sky Survey (SDSS)

Проект Sloan Digital Sky Survey (SDSS) представляет собой один из наиболее масштабных и систематизированных астрономических проектов, направленных на получение высокоточных данных о распределении, характеристиках и эволюции объектов во Вселенной. С момента своего запуска SDSS предоставил детализированные фотометрические и спектроскопические данные о миллионах галактик, звёзд, и других астрономических объектов, что делает его важным источником данных для обучения моделей машинного обучения.

SDSS предоставляет публичный API для загрузки астрономических изображений в формате FITS (Flexible Image Transport System). Так, полеченные через API изображения легли в основу набора данных для обучения рассматриваемых моделей.

B. GalaxyZoo

GalaxyZoo [11] — это общественный научный проект по классификации галактик, основанный на принципах гражданской науки. GalaxyZoo использует веб-платформу, где участникам предлагается анализировать изображения галактик, полученные в рамках проекта SDSS, и отвечать на ряд стандартизированных вопросов, таких как:

- Принадлежит ли галактика к спиральному или эллиптическому типу?
- Наблюдаются ли признаки взаимодействия с другими галактиками?
- Есть ли особенности (поперечные перемычки, кольца и др.)?

Каждое изображение классифицируется несколькими пользователями, что позволяет минимизировать ошибки. Далее данные агрегируются и проверяются с помощью статистических методов и машинного обучения.

В результате работы проекта была создана обширная база данных, содержащая информацию о классифицированных галактиках. Так, классификация GalaxyZoo состоит из 6 классов:

- **Elliptical (эллиптическая галактика)** - гладкая, симметричная форма без четкой структуры. Нет спиральных рукавов или диска.
- **Clockwise Spiral (спиральная галактика с закруткой по часовой стрелке).** Четкие спиральные рукава, закручивающиеся по часовой стрелке.
- **Anticlockwise Spiral (спиральная галактика с закруткой против часовой стрелки).** Аналогична спиральной галактике, но рукава закручены против часовой стрелки.
- **Edge-on (галактика, видимая с ребра).** Дисковая галактика (спиральная или линзовидная), наблюдаемая с края.
- **Star / Do not know (звезда / не уверен).** Объект слишком мал или размыт, может быть звездой;
- **Merger (сливающиеся галактики).** Две или более галактик в процессе слияния.

Наиболее значимыми для данной работы данными являются: наиболее вероятный класс галактики, её небесные координаты (ra, dec) и идентификатор объекта в базе данных галактик проекта SDSS.

C. Процедура создания набора данных

Основой для создания датасета послужили данные из вышеописанных источников, SDSS для изображений и GalaxyZoo для классификации объектов соответственно.

Так были загружены 5084 изображений из репозитория SDSS в FITS формате (общепринятый в астрономическом сообществе формат хранения данных), суммарно содержащие 23167 галактик. Каждый полученный файл содержит сырые снимки, полученные с ПЗС матрицы телескопа в 3 спектрах – *r, g, i,* где *r* – красный спектр, g-зеленый, i – инфракрасный.

Поэтому необходимо преобразовать данные астрономические изображения из формата FTIS в PNG с цветовой схемой RGB. Существует несколько линейных методов преобразования, однако для выделения галактик с сильными градиентами яркости линейное масштабирование не подходит. Например, радиальный профиль яркости спиральной галактики можно описать: степенным законом (профиль де Вокулёра) для балджа[1], экспоненциальным законом для диска [12]. В таких случаях обычно применяются нелинейные методы масштабирования, такие как: sinh, asinh, sqrt. Поэтому для преобразования изображений в RGB применялся метод из Lupton et al. (2004) [13], использующий arcsinh масштабирование. В частности, применялась реализация данного метода из Python-библиотеки astropy.

Получив PNG-изображения, необходимо для каждого объекта на нем задать координаты рамок (bounding boxes). Для этой задачи был разработан автоматический метjд разметки изображений. Он состоит из следующих шагов.

Первый шаг — это получение координат центров объектов на изображении, путем преобразования их небесных координат (ra, dec) в индексы пикселей на изображении. Для того использовался компонент FITS файлов WCS (World Coordinate System) и библиотека astropy.

Второй шаг – определение координат рамки объекта на основе координат его центра. Для решения этой зада-

[1] Балдж - сфероидальное уплотнение из звёзд в центре галактики.

чи был использован алгоритм, предложенный в статье [9]. Таким образом была полечена достаточно точная разметка галактик на изображениях.

В качестве финального шага, изображения были разделены на фрагменты с разрешением 512x512 пикселей. Разметка данных была оформлены в формате COCO. Итого полученный набор данных содержит 17 545 изображений. Объекты были разделены на следующие классы:

- Эллиптические галактики (p_el);
- Спиральные галактики (p_spiral);
- Галактика, видимая с ребра (p_edge);
- Неопределенный объект (p_dk);
- Сливающиеся галактики (p_mg).

Сокращенный набор данных на ~4000 изображений, использовавшийся для обучения моделей, был размещен на площадке HuggingFace[2].

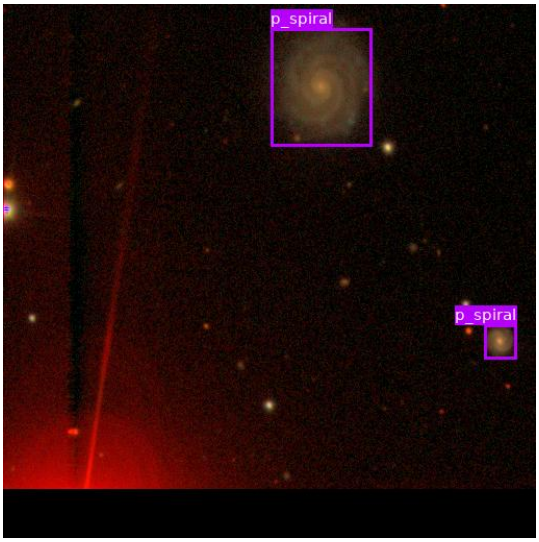


Рисунок 1. Пример размеченного изображения из набора данных

III. Нейросетевые архитектуры

A. *DETR - Detection Transformer*

DETR (DEtection TRansformer) — это нейросетевая архитектура для обнаружения объектов, основанная на трансформерной модели, которая формулирует задачу детекции как проблему прямого предсказания множества объектов.

Архитектура DETR состоит из следующих компонентов.

CNN-backbone. Исходное изображение обрабатывается с помощью CNN (например, ResNet), который создаёт карту признаков с пониженным разрешением. Эта карта служит входом для трансформера.

Трансформер энкодер. Карта признаков преобразуется в последовательность с помощью свёртки 1x1, уменьшающей количество каналов. К ней добавляются позиционные кодировки, чтобы сохранить пространственную информацию. Энкодер состоит из нескольких слоёв, каждый из которых включает self-attention механизм и полносвязную сеть (FFN). Self-attention позволяет модели анализировать глобальные взаимодействия между объектами.

Трансформер декодер. Декодер следует стандартной архитектуре трансформера, преобразуя N эмбеддингов размерности d с использованием механизмов self-attention и encoder-decoder attention. В отличие от оригинального трансформера, DETR декодирует N объектов параллельно на каждом слое декодера.

Декодер также инвариантен к перестановкам, N входных эмбеддингов должны быть различными, чтобы генерировать разные результаты. Эти входные эмбеддинги представляют собой обучаемые позиционные кодировки, которые называются object queries (запросы объектов). N запросов объектов преобразуются декодером в выходные эмбеддинги, которые затем независимо декодируются полносвязной сетью.[14]

Feed-forward networks (FFNs). Окончательное предсказание формируется трёхслойным перцептроном с функцией активации ReLU и скрытой размерностью d, за которым следует линейный проекционный слой. FFN предсказывает нормализованные координаты центра, высоту и ширину bounding box относительно входного изображения, а линейный слой определяет метку класса с помощью функции softmax. [14]

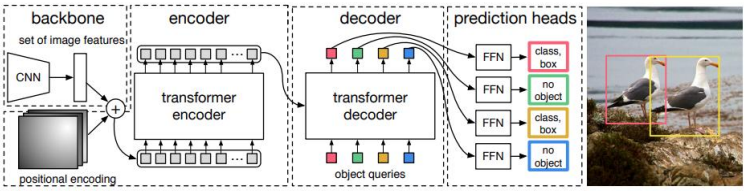


Рисунок 2. Схема архитектуры DETR

B. *YOLOv11 (You Look Only Once)*

Архитектура YOLO (You Only Look Once) представляет собой серию моделей для детектирования объектов в реальном времени, отличающихся высокой скоростью и точностью.

В основе архитектуры YOLO лежит идея обработки всего изображения за один проход нейронной сети, что позволяет одновременно предсказывать координаты ограничивающих рамок и соответствующие им классы объектов. Это достигается благодаря использованию свёрточных нейронных сетей (CNN), которые разделяют изображение на сетку и для каждой ячейки предсказывают несколько ограничивающих рамок и вероятности принадлежности выделенных объектов к определенному классу.

Архитектура YOLO состоит из трех фундаментальных компонентов:

- **Backbone (основная сеть)**: отвечает за извлечение признаков из входного изображения. Этот процесс включает в себя наложение сверточных

[2] Набор данных на HuggingFace - https://huggingface.co/datasets/0berheim/GalaxyDetectionDataset

слоев для генерации карт признаков в различных разрешениях.

- **Neck (промежуточная сеть)**: служит для объединения признаков с разных уровней пирамиды признаков, что позволяет эффективно детектировать объекты различных размеров.
- ***Head* (или detect, выходной слой)**: Предназначен для предсказания координат ограничивающих рамок, классов объектов и соответствующих вероятностей. В более поздних версиях YOLO были внедрены механизмы, такие как Decoupled Head, для раздельного предсказания классификации и регрессии, что улучшило точность модели.

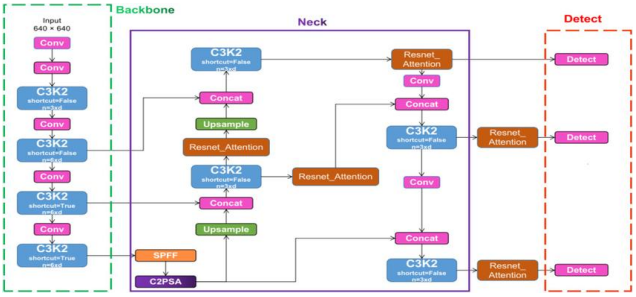


Рисунок 3. Схема архитектуры YOLOv11

Для решения поставленной задачи использовалась последняя версия архитектруы - YOLOv11. Данная версия включает несколько ключевых архитектурных улучшений, направленных на повышение точности и эффективности модели.

Рассмотрим основные нововведения в YOLOv11:

- **Блок C3k2:** Данный блок представляет собой сверточный слой, использующийся в компоненте Backbone для извлечения признаков изображения. C3k2 является заменой блока C2f из прошлых версий архитектуры и демонстрирует более высокую вычислительную эффективность. Он использует две свертки малого размера вместо одной большой, как это было в YOLOv8. “k2” в названии означает меньший размер ядра равный 2, что способствует более быстрой обработке при сохранении производительности [15].
- **SPPF (Spatial Pyramid Pooling - Fast):** Модуль *SPPF* обеспечивает многомасштабное объединение признаков, что способствует лучшему распознаванию объектов различных размеров и форм.
- **C2PSA (Convolutional block with Parallel Spatial Attention)**: Этот свёрточный блок с параллельным пространственным вниманием усиливает способность модели фокусироваться на значимых областях изображения, улучшая детектирование объектов в сложных сценах.

C. RT-DETR (You Look Only Once)

RT-DETR (Real-Time Detection Transformer) — это передовой детектор объектов, разработанный Baidu, который обеспечивает производительность в реальном времени, сохраняя при этом высокую точность. Основой модели является архитектура Vision Transformer, с целью использования для задач обнаружения объектов с акцентом на скорость обработки. RT-DETR устраняет необходимость в отдельных этапах для предложения и классификации объектов, упрощая общий конвейер и потенциально повышая эффективность.

RT-DETR использует CNN-backbone (Рисунок 4) для формирования карт признаков исходного изображения. Эффективный гибридный кодировщик обрабатывает разномасштабные признаки, разделяя внутримасштабное взаимодействие и межмасштабное слияние. Такой подход позволяет увеличить точность предсказания и эффективность работы нейронной сети. Кодировщик состоит из двух основных компонентов: межмасштабного слияния признаков на основе CNN (CCFM) и взаимодействия внутримасштабных признаков на основе внимания (AIFI). Transformer decoder с intra-scale feature interaction module итеративно оптимизирует object queries для получения ограничивающих рамок и оценок достоверности.

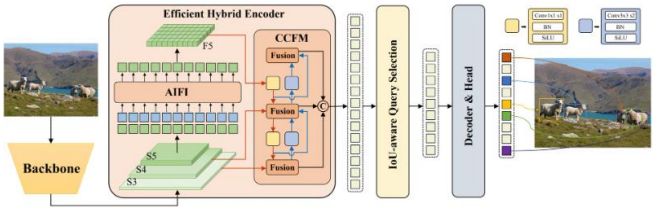


Рисунок 4. Схема архитектуры RT-DETR

IV. Сравнение

Полученный датасет в 17 тысяч изображений был сокращен в 4 раза. Это было сделано с целью ускорения процесса обучения и уменьшения влияния дисбаланса классов, поскольку во вселенной одни типы космических объектов встречаются значительно чаще остальных, а некоторые явления, например, слияние галактик – происходят совсем редко. Итоговый датасет делился на три части: тренировочный, тестовый, валидационный в соотношении 0.7/0.2/0.1 соответственно.

Поскольку скорость обучения и работы YOLO значительно превышает ту, что предоставляют модели типа DETR, для второй нейросети было решено двукратно увеличить число эпох, чтобы время обучения было примерно равным (около двух часов). На наш взгляд, основными критериями для сравнительного анализа нейронных сетей, должны являться скорость работы/обучения, а также точность определения объектов, поэтому именно эти критерии будут наиболее подробно рассматриваться.

Нейросети архитектуры DETR более требовательными к вычислительным мощностям, что вынудило нас производить обучение на 45 эпохах. Однако, после примерно 30-й эпохи нейросеть практически не улучшала результат своей работы, что говорит одновременно и об отсутствии переобучения и недообучения (Рисунок 5). Данное явление объясняется и особенностями набора данных, что будет рассматриваться ниже. Можно заметить сильное колебание cardinality error, что вероятно, и является ключевой причиной, по которой после определенной эпохи точность нейросети стагнировала.

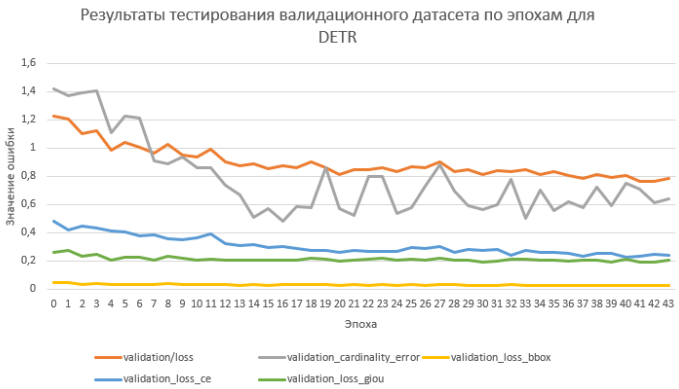


Рисунок 5. Результаты обучения нейросети DETR в течение 45 эпох

Далее обучение производилось с использованием YOLOv11 на 100 эпохах, поскольку каждая итерация выполнялась примерно за минуту. Дополнительно к этому данная нейронная сеть является наиболее быстрой из всех перечисленных, что, несомненно, выделяет ее на фоне других. Однако в задачах детекции типов галактик данный показатель является скорее приятным дополнением, а не ключевой метрикой, поскольку оборудование вероятно будет установлено стационарно, то и проблемы с вычислительной мощностей являются некритичными. В отличии от DETR данная модель не так сильно стагнировала в конце обучения и, вероятно, при увеличении числа эпох был бы продемонстрирован более качественно результат, однако на графиках видно, что уменьшение числа ошибок сильно замедлилось (Рисунок 6).

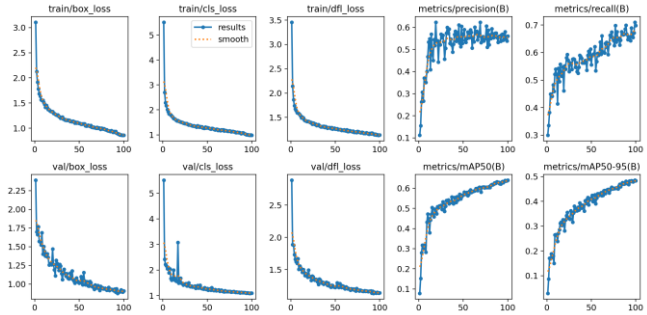


Рисунок 6. Результаты обучения нейросети YOLOv11 в течение 100 эпох

Последней нейронной сетью, обученной на этом датасете, стала RT-DETR. Обучение происходило в течение 45 эпох, поскольку скорость работы была сопоставима с той, что демонстрировал простой DETR. Как видно из графиков (Рисунок 7), аналогично с YOLOv11 стагнация не так заметна, метрики валидационной выборки методично улучшались даже на последних этапах обучения. Это может говорить, что существует возможность улучшения результата предсказаний при банальном увеличении числа эпох.

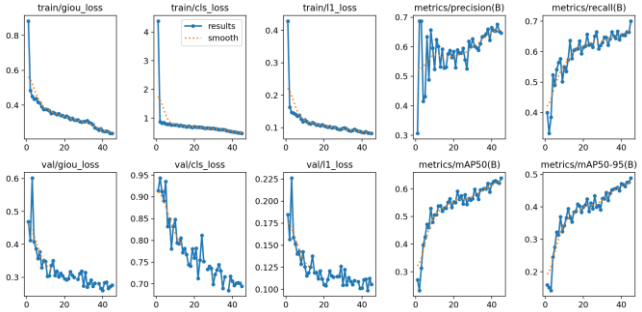


Рисунок 7. Результаты обучения нейросети RT-DETR в течение 45 эпох

Рассмотрим матрицу точности в предсказании классов, которая получилась по итогам обучения RT-DETR (Рисунок 8). Как видно из рисунка, нейросеть часто обнаруживает объект, который отсутствует на изображении или был не размечен, особенно для классов p_el и p_dk. Другие же классы, отличные от фона, практически не спутываются, а даже если и подменяются, то на порядки раз меньше, чем в случае с детекцией неразмеченного объекта.

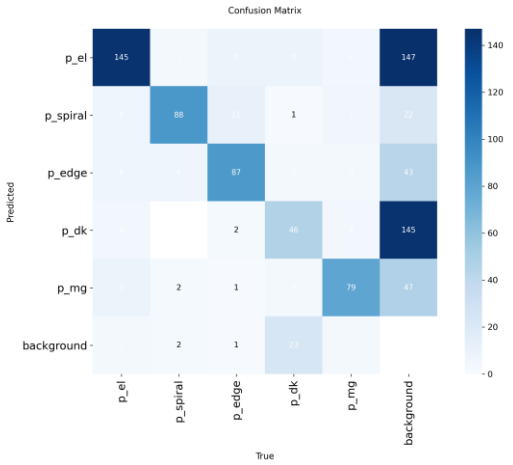


Рисунок 8. Матрица точности определния класса с использованием RT_DETR (больше вне диагонали – хуже)

Изучая результаты выполнения, можно заметить, что датасет, размеченный автоматическим способом на основе данных полученных из проекта GalaxyZoo, содержит неоднозначность и неполноту в выделении объектов. Так, например, слияние галактик может быть выделено двумя отдельными bbox'ами. Артефакты, представляющие собой траектории околоземных объектов (спутники, астероиды), снятые через зеленый фильтр, выделяются лишь единожды в случайном месте луча. В подобных изображениях нейронные сети зачастую находили несколько объектов с небольшой степенью уверенности, что сильно влияло на статистику (рисунок 9).

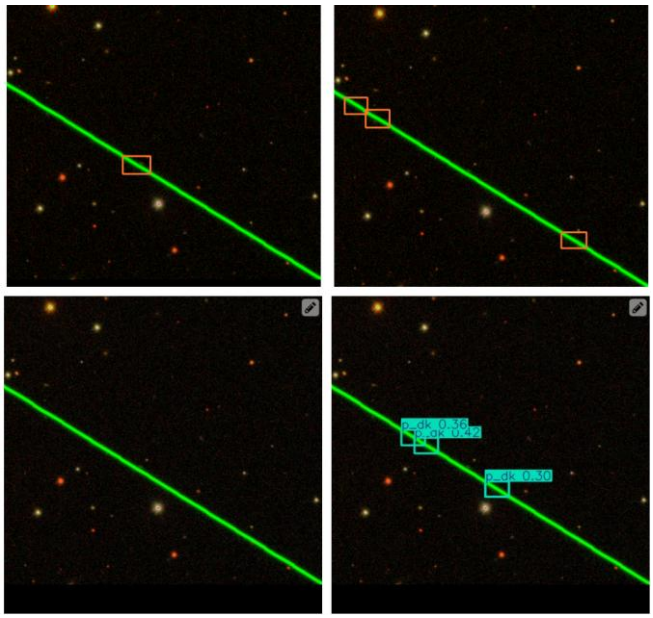


Рисунок 9. Результат детекции артефакта (p_dk) различными моделями (слева вниз): оригинал, DETR, YOLOv11, RT-DETR.

Для DETR минимальная уверенность при детекции была выставлена в 0.25, только так получалось получить хотя бы один объект на изображении, иначе складывалась ситуация, схожая с YOLO. Подобных артефактов в датасете содержится достаточно много, и иногда данные лучи имеют большую длину, что сильно ухудшало cardinality error, а также матрицу точности.

В связи с этим, часть датасета была скорректирована нами в ручном режиме, однако поскольку число изображений измеряется тысячами, а нейросети показывали хороший результат на основе визуального анализа (рисунок 10), а также нами была выбрана тема в изучении возможности валидации. Было решено остановиться на 500 проверенных или измененных изображениях, которые попали в итоговый датасет.

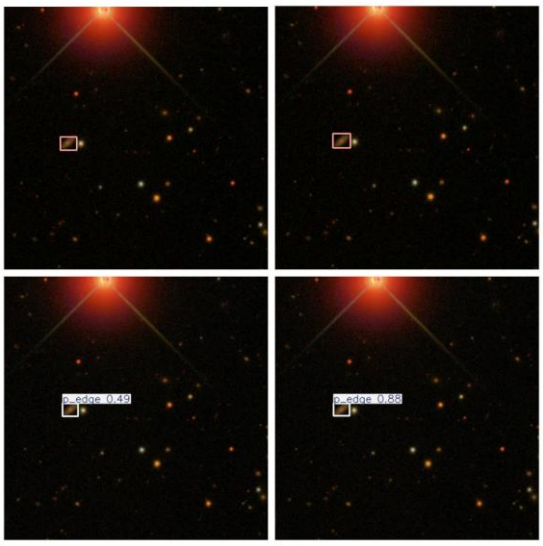


Рисунок 10. Результат детекции торца галактики (p_edge) различными моделями (слева вниз): оригинал, DETR, YOLOv11, RT-DETR.

Исходя их того, что оригинальная разметка составлялась с использованием автоматического подхода, она является неполноценной, как можно заметить на наглядном примере (рис. 11). На изображении видно, что в оригинале было всего два объекта, однако RT-DETR нашел все действительно существующие объекты и выделил их. YOLOv11 и DETR разметили дополнительно по одному объекту.

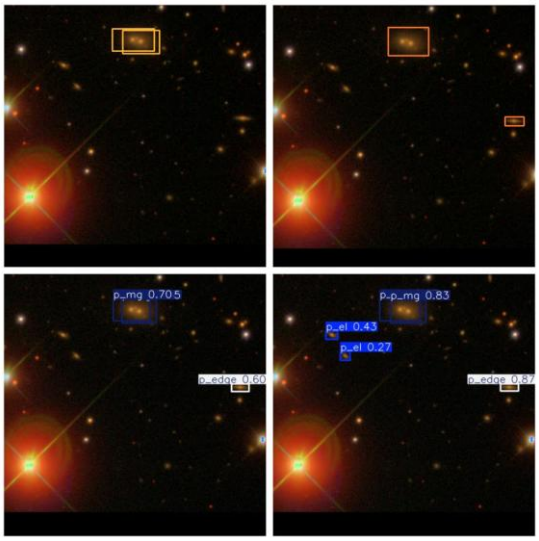


Рисунок 11. Результат детекции случайного изображения (5482.png) различными моделями (слева вниз): оригинал, DETR, YOLOv11, RT-DETR.

Подобная ситуация наблюдалась практически на любом изображении: либо все оригинальные объекты были найдены, либо была произведена детекция неразмеченных объектов (рис. 11). На основе визуального анализа, получается, что ошибки первого рода практически редко встречаются при использовании DETR или YOLOv11 (но реже чем у первой модели), а для RT_DETR практически отсутствуют. Настоящие ошибки второго рода крайне редко встречались у всех трех нейросетей, однако наличие «фальшивых» результатов было регулярным.

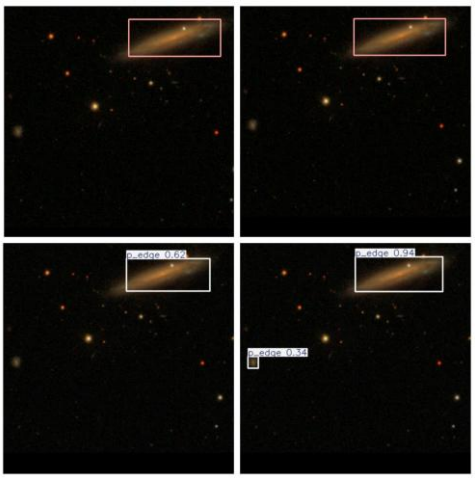


Рисунок 12. Результат детекции случайного изображения (9647.png) различными моделями (слева вниз): оригинал, DETR, YOLOv11, RT-DETR.

V. Заключение

В работе было произведено сравнение различных архитектур нейронных сетей с целью выделения наиболее подходящего кандидата на роль полноценного детектора галактик, способного убрать субъективный фактор, а также ускорить и автоматизировать процесс определения типа объектов.

По результатам сравнения нейронных сетей было выявлено, что собранный датасет нуждается в доработке и ручном исправлении ошибок в полном виде. Это позволит сделать метрики более объективными, что придаст выбору подходящей модели фактическую основу. Также, предположительно, датасет является достаточным по размеру и больше 800 изображений на класс является избыточным числом.

На наш взгляд, наиболее подходящим выбором является модель типа RT-DETR, потому что данная нейронная сеть продемонстрировала наилучший результат, как по метрикам, так и по визуальному анализу результатов работы сети. Данная нейросеть включает в себя сильные стороны других кандидатов, что делает ее предсказания наиболее точными.

Для улучшения эффективности работы нейронной сети могут быть сделаны следующие шаги: расширенный подбор гиперпараметров, увеличение числа эпох обучения, а также полная корректировка датасета. Эти шаги должны сильно повысить качество работы модели, что позволит использовать её для реальной работы.

По результатам работы нейросетевых решений можно утверждать, что данная задача успешно решается при помощи методов глубокогообучения и компьютерного зрения, что открывает возможность автоматизации или упрощения процесса классификации космических изображений. Рассмотренные в работе методы являются популярными и относительно новыми подходами к детекции объектов.

Литература

[1] Sophya Belyakova. (2025). Особенности детектирования знаков дорожного движения «Пешеходный переход».

[2] Altunian, Alisa & Kiselev, Stanislav. (2025). Локальные методы планирования траекторий на основе обучения с подкреплением.

[3] Katyzina, Anastasia & Fam, Anastasia. (2025). Нейросетевое распознавание и мониторинг состояния водоемов на спутниковых изображениях.

[4] Vasileva, Anna & Grimm, Matvey. (2025). Применение компьютерного зрения для детекции загрязненных зон пляжей.

[5] Krivorot, Iuliia & Ilyakov, Egor. (2025). Детектирование диких животных при помощи нейронных сетей.

[6] Dubath, Pierre & Apostolakos, Nikolaos & Bonchi, Andrea & Belikov, A. & Brescia, Massimo & Cavuoti, Stefano & Capak, Peter & Coupon, Jean & Dabin, Christophe & Degaudenzi, Hubert & Desai, Shantanu & Dubath, Florian & Fontana, Adriano & Fotopoulou, Sotiria & Frailis, Marco & Galametz, Audrey & Hoar, John & Holliman, Mark & Hoyle, Ben & Zacchei, Andrea. (2017). The Euclid Data Processing Challenges. Proceedings of the International Astronomical Union. 12. 10.1017/S1743921317001521.

[7] Pöntinen, M., et al. "Euclid: Identification of asteroid streaks in simulated images using deep learning," in Astronomy & Astrophysics, vol. 679, pp. A135, 2023.

[8] Schaefer, C., et al. "Deep convolutional neural networks as strong gravitational lens detectors," in Astronomy & Astrophysics, vol. 611, pp. A2, 2018.

[9] Z. He, B. Qiu, A. -L. Luo, J. Shi, X. Kong and X. Jiang, "Deep learning applications based on SDSS photometric data: detection and classification of sources," in Monthly Notices of the Royal Astronomical Society, vol. 508, no. 2, pp. 2039-2052, Sept. 2021, doi: 10.1093/mnras/stab2243.

[10] Xiao, Yao & Guo, Yang & Pang, Qinghao & Yang, Xu & Zhao, Zhengxu & Yin, Xianlong. (2025). STar-DETR: A Lightweight Real-Time Detection Transformer for Space Targets in Optical Sensor Systems. Sensors. 25. 1146. 10.3390/s25041146.

[11] Lintott, C., Schawinski, K., Slosar, A., Land, K., Bamford, S., Thomas, D., Raddick, M., Nichol, R., Szalay, A., Andreescu, D., Murray, P., and Vandenberg, J. 2008. Galaxy Zoo: morphologies derived from visual inspection of galaxies from the Sloan Digital Sky Survey. Monthly Notices of the Royal Astronomical Society, 389(3), p.1179‑1189.

[12] Roberto E. González, Roberto P. Muñoz, & Cristian A. Hernández. (2018). Galaxy detection and identification using deep learning and data augmentation.

[13] Lupton, R., Blanton, M., Fekete, G., Hogg, D., O'Mullane, W., Szalay, A., and Wherry, N. 2004. Preparing Red-Green-Blue Images from CCD Data. The Publications of the Astronomical Society of the Pacific, Volume 116, Issue 816, pp. 133-137.

[14] Nicolas Carion., et al, "End-to-End Object Detection with Transformers," 2020.

[15] Rahima Khanam and Muhammad Hussain. YOLOv11: An Overview of the Key Architectural Enhancements. arXiv: 2410.17725v1, 2024. Available at: https://arxiv.org/abs/2410.17725.

ГРНТИ: 28.23.15, 67.01.29, 67.03.03
ASJC: 1707, 2215, 2216
OECD: 01.02.EP

Сегментация строений на изображениях с видом сверху

Р. А. Каримов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2410746@edu.misis.ru

М. Э. Насибов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2005329@edu.misis.ru

***Аннотация* — В данной работе рассматриваются различные подходы для решения задачи сегментации строений — от классических архитектур до современных трансформеров, а также проводится их сравнение по точности и скорости работы. Особое внимание уделяется адаптации моделей к разным условиям съемки и качеству изображений. Результаты экспериментов подтверждают, что глубокое обучение позволяет надежно детектировать здания даже на сложных снимках с помехами.**

Ключевые слова — Нейронные сети, Сегментация изображений, Вид сверху, Строения, Обработка изображений, SAM, Swin Transformer, Mask2Former.

I. Введение

Извлечение зданий из изображений дистанционного зондирования с высоким пространственным разрешением предоставляет важную информацию для городских приложений, таких как умные города, градостроительство, оценка численности населения и управление в условиях чрезвычайных ситуаций [1-4]. С тех пор, как Liow и др. [5] начали изучать автоматическое извлечение зданий с аэроснимков в 1989 году, было предложено множество известных алгоритмов.

В изображениях высокого разрешения часто наблюдаются значительные внутриклассовые различия при относительно небольших межклассовых различиях. Кроме того, снимки дистанционного зондирования подвержены влиянию изменений освещения, рельефа, окружающей среды и атмосферных условий. Эти особенности крайне затрудняют точное извлечение признаков из изображений высокого разрешения.

Здания обладают большим разнообразием типов и форм, а их размеры, как правило, меньше по сравнению с другими объектами, такими как дороги и водные пространства, что дополнительно усложняет их автоматическое выделение. Таким образом, современные алгоритмы извлечения зданий требуют дальнейшего совершенствования.

Современные подходы к выделению зданий включают контролируемые, слабо контролируемые и неконтролируемые методы. Первые требуют большого количества размеченных данных, что ограничивает их применение. Альтернативные подходы используют псевдометки или работают без разметки, но сталкиваются с проблемами точности [6]. Перспективным направлением являются фундаментальные модели вроде Segment Anything Model (SAM), однако их эффективность для зданий остаётся неидеальной [7].

Таким образом, задача автоматического выделения зданий требует новых решений. Комбинирование современных моделей с дополнительными методами обработки может улучшить качество сегментации при работе с реальными данными. Это особенно важно для практических приложений в урбанистике и мониторинге.

II. Набор данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей был создан собственный набор данных, включающий в себя более 2000 изображений строений, сделанных на карте со спутника из бесплатных сервисов [8].

Набор данных был создан с помощью программного кода на Python, который загрузил карту с помощью сервиса Mapbox. В программный код были введены координаты различных городов, а также различные трансформации для преобразования изображений, в частности различные повороты исходных изображений в диапазоне от -20 до +20 градусов. Набор данных разделён на обучающую, валидационную и тестовую выборки в соотношении 85/10/5, а именно 1732/210/103 изображений соответственно.

Из-за различий в структуре городов, на изображениях можно найти различные элементы - пригородные строения, водоемы, леса, поля, парки, мосты, склады и другие структуры присущие определённому городу. Помимо изображений с определёнными строениями, были также получены изображения, которые вовсе не включают в себя ни одного здания (координаты попали в водоем, либо другую местность, на которой отсутствуют строения).

Конечно, различие в снимках может и помешать - с одной стороны, обученная модель сможет различать строения различной структуры и размера, а с другой, сильное различие и несбалансированность могут

помешать при обучении, ведь существуют различные факторы сделанных снимков со спутника (текущее время, тени от солнца и т.д.) (рисунок 1).

Рис. 1. Примеры снимков различного вида

III. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. *SAM + Canny Edge Detector*

В работе [9] авторы предложили инновационный подход к сегментации зданий на спутниковых и аэрофотоснимках, полностью без использования размеченных данных. Их метод основан на комбинации двух ключевых технологий - SAM и Canny Edge Detector.

Основная идея метода заключается в использовании SAM — мощной модели от Meta AI, способной сегментировать объекты по запросу или в автоматическом режиме. Однако прямое применение SAM к аэрофотоснимкам приводит к избыточной сегментации (одно здание разбивается на несколько масок) и пропускам объектов [10].

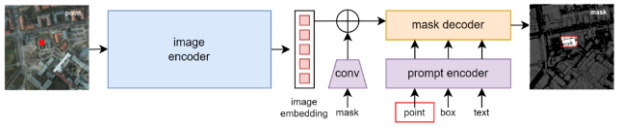


Рис. 2. Структура SAM с точечной подсказкой и соответствующей маской

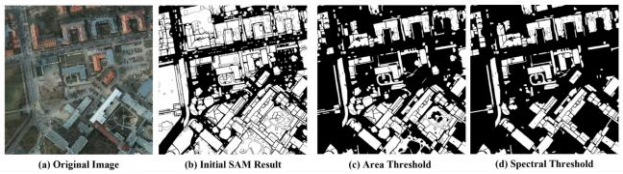


Рис. 3. Последовательное маскирование полигонов: (a) Исходное изображение, (b) Исходные маски SAM, сгенерированные в режиме автоподсказки, (c) Результат после применения порогового значения на основе площади, (d) Результат после применения порогового значения на основе NDVI и BAI.

Для улучшения качества авторы предложили двухэтапную фильтрацию результатов SAM:

- Отбор по площади — удаление слишком крупных (например, участков леса) и мелких объектов (шумов).
- Классификация с помощью спектральных индексов (NDVI, BAI) — выделение зданий и устранение ложных срабатываний (дороги, растительность) [11].

$$NDVI = \frac{NIR-Red}{NIR+Red}, \quad (1)$$

$$BAI = \frac{Blue-NIR}{Blue+NIR}, \quad (2)$$

где Blue, Red и NIR означают значения яркости соответствующих диапазонов: синего, красного и ближнего инфракрасного.

Затем каждый объект классифицировался на основе эвристически определенного порогового значения. Понимая, что эвристический подход к определению порога может внести в модель человеческое вмешательство и предвзятость, авторы смягчили эту проблему, установив порог как можно ниже, тем самым минимизировав комиссионные ошибки. Учитывая, что данная модель работает на основе слабого контролируемого обучения и использует информацию о краях для различения объектов, она не полностью полагается на псевдометки и по своей сути устойчива к несовершенным классам меток пикселей и пороговым значениям. Поэтому смещение, вносимое классами пикселей при эвристическом подходе, может быть в определенной степени проигнорировано.

Учитывая эффективность спектрального индекса, сначала были выделены классы зданий, а затем последовательно удалены растительность и дороги (рисунок 4).

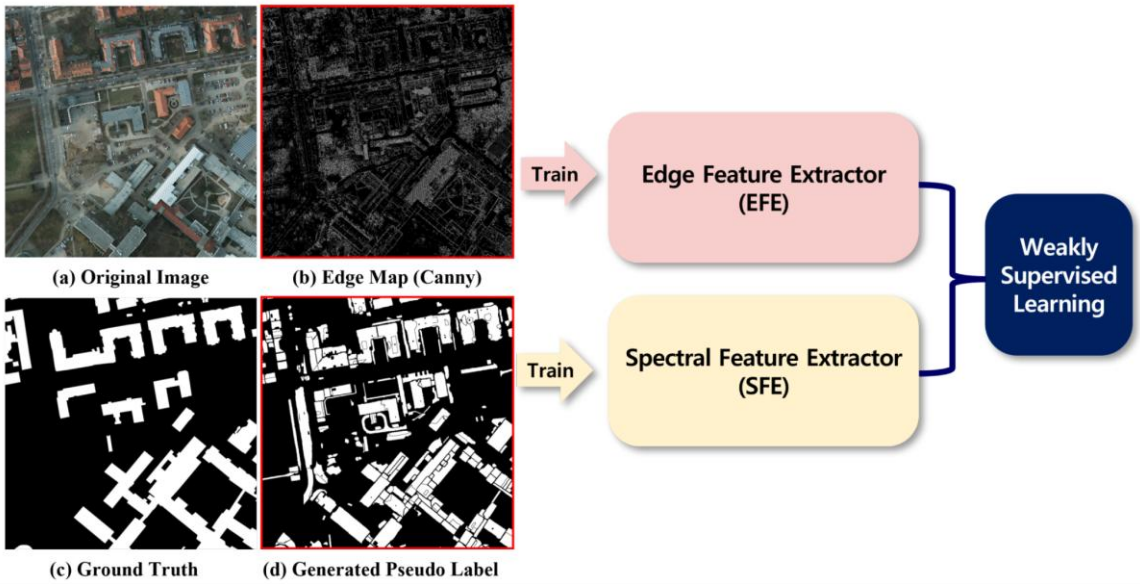


Рис. 4. Результаты после неконтролируемого извлечения признаков: (a) исходное изображение, (b) результат после применения детектора краев Canny, (c) истинное изображение для сравнения, (d) сгенерированная псевдометка.

Несмотря на фильтрацию, полученные псевдо-метки остаются неточными: границы зданий размыты, присутствуют артефакты. Чтобы исправить недостатки псевдо-меток, авторы разработали специальную нейросетевую архитектуру, состоящую из двух модулей:

- Spectral Feature Extractor (SFE) — анализирует спектральные характеристики пикселей, классифицируя их как "здание" или "фон".
- Edge Feature Extractor (EFE) — использует карту границ (полученную детектором Канни) для восстановления четких контуров зданий [12].

Модель обучается без участия человека, используя только псевдо-метки и граничную информацию. Для финального уточнения применяется CRF (Conditional Random Fields) [13], что позволяет сгладить шумы и улучшить границы объектов.

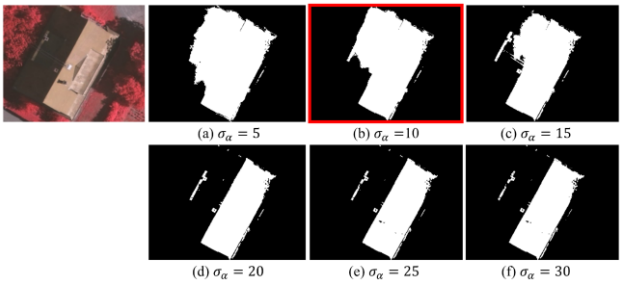


Рис. 5. Тонкая подгонка CRFs. Красная рамка представляет собой наилучший результат.

Метод тестировался на стандартных датасетах ISPRS Potsdam и Vaihingen и показал следующие результаты: F1-score = 0.7829, IoU = 0.6463 (Potsdam) [14], что превосходит другие unsupervised-методы (IIC, STEGO, HP). В supervised-режиме (с дообучением на размеченных данных) модель демонстрирует точность, близкую к MANet и ResUNet, но с лучшей детализацией границ.

Подход универсален — аналогичная методика применима для сегментации дорог и растительности. Предложенный метод открывает новые возможности для автоматического анализа спутниковых данных без ручной разметки. Однако остаются проблемы:

- Недооценка площади зданий (низкий Recall из-за избыточной фильтрации).
- Чувствительность к теням (SAM иногда выделяет их как отдельные объекты).
- Соль-и-перечный шум на выходных масках.

В будущих работах авторы планируют улучшить обработку текстур крыш и адаптировать метод для работы в условиях облачности и затенения. Работа демонстрирует, что комбинация SAM и edge-информации позволяет эффективно решать задачу сегментации зданий без привлечения размеченных данных. Подход особенно актуален для задач, где ручная разметка невозможна или экономически невыгодна, например, при анализе больших территорий или работе с историческими снимками.

B. Mask2Former + Swin Transformer

Второй моделью для решения задачи является Mask2Former в сочетании с магистральной сетью Swin Transformer [15]. Данная комбинация представляет собой универсальную архитектуру сегментации изображений, основанную на простой мета-архитектуре (рисунок 6), содержащий магистральную сеть, пиксельный декодер и трансформирующий декодер. Средство извлечения основных функций может быть построено на основе моделей на базе CNN (т.е. ResNet) или transformer.

В отличие от MaskFormer [16], который использует сеть пирамид признаков [17] в качестве пиксельного декодера Mask2Former использует многомасштабный деформируемый преобразователь внимания (MSDeformAttn) в качестве пиксельного декодера по умолчанию [18] для включения функций с низким и высоким разрешением при одновременном ограничении роста вычислений.

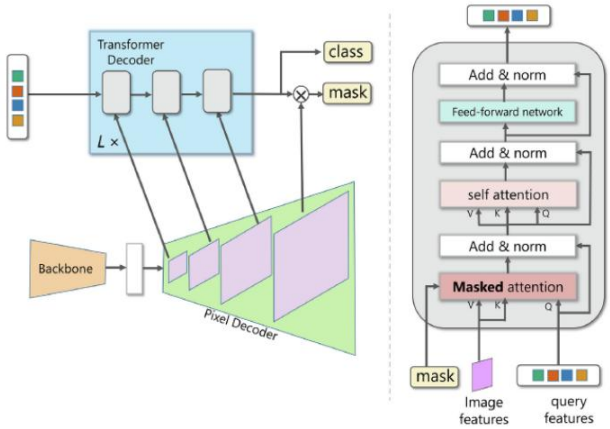


Рис. 6. Архитектура Mask2Former. В качестве Backbone (магистральной сети) будет выступать Swin Transformer

Swin transformer - мощный визуальный преобразователь для извлечения многомасштабных характеристик из данных и захвата зависимостей на больших расстояниях. За счет использования технологии Shifted Windows [19], которая ограничивает самостоятельные вычисления неперекрывающимися локальными окнами и облегчает межоконные соединения [20], повышается общая эффективность системы. Архитектура Swin transformer состоит из четырех этапов: разделение патчей [21], объединение патчей, линейное встраивание и блоки Swin transformer (рисунок 7).

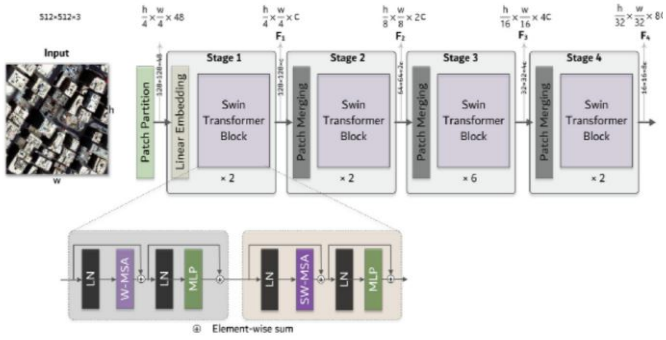


Рис. 7. Обзор структуры работы Swin Transformer с входным изображением

Принцип работы (end-to-end):

1. Вход: Изображение размером H×W×3 (спутниковый снимок).
2. Swin Transformer:
   - Извлекает признаки 4 уровней (C1–C4).
3. Pixel Decoder:
   - Создаёт карты признаков высокого разрешения.
4. Mask2Former Decoder:
   - Шаг 1: Инициализация объектных запросов.
   - Шаг 2: Для каждого запроса:
     - Self-attention между запросами [20].
     - Masked cross-attention с картами признаков (фокус на релевантных пикселях).
     - FFN → предсказание маски и класса.
   - Шаг 3: Уточнение масок через несколько слоёв декодера.
5. Выход:
   - N масок (H/4×W/4) + метки классов.
   - Постобработка: Разрешение масок повышается до H×W билинейной интерполяцией [22].

В этом исследовании оценивалось качество моделей глубокого обучения на датасетах ISPRS Potsdam и Vaihingen с использованием различных метрик, включая среднее пересечение по объединению (IoU) и F1-score. по этим метрикам данная модель достигла значений IoU = 0.8827 F1-score = 0.9367.

В ходе работы, Mask2Former + Swin Transwormer показала высокую эффективность в мультимасштабности, высоком разрешении, точности границ, но при этом это решение имеет относительно невысокие вычислительные требования. Также, тонкая настройка гиперпараметров может повысить обобщаемость и переносимость модели в широком диапазоне географических местоположений и условий окружающей среды, значительно повысив ее пригодность для многочисленных городских приложений. Кроме того, производительность модели может быть улучшена за счет включения дополнительных наборов данных, в том числе мультиспектральных каналов и данных лазерного сканирования.

IV. Сравнение

Сравним два описанных подхода. Для сравнения используется собственный набор данных, содержащий 2044 изображений строений [23]. Датасет был поделен на обучающую (1732 снимок), валидационную (210 снимков) и тестовую (103 снимка) выборки. Для расчета качества работы моделей используется пиксельный подход, так как сегментация — это задача классификации каждого пикселя изображения. Рассмотрим процесс на примере сегментации зданий:

Введём следующие определения:

- Ground Truth (GT): Бинарная маска, где:
  - 1 (белый) — пиксель принадлежит целевому классу (напр., здание),
  - 0 (чёрный) — фон.

- Predicted Mask: Бинарная маска, созданная моделью (значения 1/0).

Для расчета метрик потребуются величины TP, TN, FP, FN для каждого пикселя изображения:

- TP – Модель верно выделила здание (GT = 1 и Prediction = 1);
- TN – Модель верно идентифицировала фон (GT = 0 и Prediction = 0);
- FP – Модель ошибочно назвала фон зданием (GT = 0 и Prediction = 1);
- FN – Модель пропустила здание, то есть не выделила (GT = 1 и Prediction = 0).

По введённым величинам строятся такие функции оценок, как:

- $Precision = \frac{TP}{TP+FP}$ (1) – Доля верно выделенных объектов среди всех предсказанных;
- $Recall = \frac{TP}{TP+FN}$ (2) – Доля найденных объектов среди всех истинных;
- $F1 = 2\frac{Precision \cdot Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN}$ (3) – оценка баланса между точностью (precision) и полнотой (recall).

Оценка сегментации также производится при помощи расчёта меры Жаккара (Intersection over Union, IoU):

$$IoU = \frac{TP}{TP + FP + FN} \quad (4)$$

Таблица 1 отображает количественные оценки для двух подходов.

ТАБЛИЦА 1. Оценка работы моделей

Модель	**SAM + Canny Edge Detector**		**Mask2Former + Swin Transformer**	
Выборка	Обучающая	Валидационная	Обучающая	Валидационная
TP	608	62	759	79
TN	662	94	723	101
FP	153	33	131	16
FN	324	21	119	14
Precision	0.8	0.66	0.85	0.83
Recall	0.65	0.75	0.87	0.85
F1	0.72	0.7	0.86	0.85
IoU	0.57	0.54	0.76	0.74

Как видно из таблицы, связка Mask2Former + Swin Transformer имеет значительно превосходящие показатели, она также показала более высокие значения на тестовой выборке: F1 = 0.88 и IoU = 0.78, в то время как у SAM + Canny метрики F1 и IoU равны 0.72 и 0.56 соответственно, что означает, что она намного лучше выделяет здания на изображениях.

Численные оценки качества сегментации нейросети имеют значения, намного ближе к 1, в отличие от первой модели, где значение IoU 0.59. Такое отличие, в сравнение с предыдущей нейронной сетью в качестве можно объяснить несколькими причинами: SAM + Canny имеет более сильную зависимость от четкости границ, а также имеет проблемы с текстурированными областями и пропуском теней. При этом второй подход более устойчив к шумам, лучше учитывает архитектурные особенности и имеет более высокую топологическую сохранность.

Таким образом, сравнивая модели SAM + Canny и Mask2former + Swin Transformer, можно обнаружить, что при использовании первого подхода затрачивается меньше вычислительных ресурсов и времени. Можно сделать вывод, что связку SAM + Canny следует использовать в условиях, когда нам предоставлены данные высокого разрешения и у нас ограниченное количество вычислительных ресурсов.

V. Заключение

Были рассмотрены и проверены на собственном наборе данных 2 модели нейронной сети. Приведены два подхода к задаче – комбинация SAM от Meta AI с детектором границ Canny и нейросетевое решение Mask2Former с магистральной сетью Swin Transformer. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведённые подходы были сравнены на разработанном наборе данных. Было оценено качество сегментации строений. По полученным данным очевидно, что подход Mask2Former + Swin Transformer, подготовленный авторами работы [24], имеет сильное преимущество перед альтернативным подходом в отношении точности работы сегментации и адаптируемости к специфике данных, что объясняется нечеткими границами объектов и низким контрастам. При этом подход с использованием SAM требует меньше вычислительных ресурсов и меньше зависит от количества размеченных данных, а также обладает большей скоростью вывода.

Литература

[1] Кожаринов, А. С. Первоочередные направления и задачи развития интеллектуальной системы прогнозирования состояний зданий и сооружений / А. С. Кожаринов // Новая наука: От идеи к результату. – 2017. – № 1-2. – С. 153-157. – EDN XQVPXV.

[2] Jianfeng Huang, Xinchang Zhang, Qinchuan Xin, Ying Sun, Pengcheng Zhang, (2019). Automatic building extraction from high-resolution aerial images and LiDAR data using gated residual refinement network, ISPRS Journal of Photogrammetry and Remote Sensing, Volume 151, Pages 91-105, ISSN 0924-2716, https://doi.org/10.1016/j.isprsjprs.2019.02.019.

[3] Sun, G., Huang, H., Zhang, A., Li, F., Zhao, H., & Fu, H. (2019). Fusion of Multiscale Convolutional Neural Networks for Building Extraction in Very High-Resolution Images. Remote Sensing, 11(3), 227. https://doi.org/10.3390/rs11030227.

[4] Рамзайцев Д.А., Матяш Д.С. Исследование возможности распознавания объектов на спутниковых снимках / Д.А. Рамзайцев, Д.С. Матяш // НИТУ МИСиС – Сборник статей

научно-технического семинара студентов кафедры «Инженерной кибернетики» – Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях – 2023 – С. 127-132.

[5] Yuh-Tay Liow, Theo Pavlidis, 1990. Use of shadows for extracting buildings in aerial images, Computer Vision, Graphics, and Image Processing, Volume 49, Issue 2, Pages 242-277, ISSN 0734-189X, https://doi.org/10.1016/0734-189X(90)90139-M.

[6] Van Etten, A.; Lindenbaum, D.; Bacastow, T.M. Spacenet, (2018)/ A remote sensing dataset and challenge series. arXiv:1807.01232.

[7] Kirillov, A.; Mintun, E.; Ravi, N.; Mao, H.; Rolland, C.; Gustafson, L.; Xiao, T.; Whitehead, S.; Berg, A.C.; Lo, W.Y.; et al, (2023). Segment anything. arXiv:2304.02643.

[8] Лычагов, А. С. Анализ доступных способов получения картографической информации в целях навигации наземных подвижных объектов / А. С. Лычагов, Р. Н. Садеков, Д. В. Яковлев // Труды ФГУП "НПЦАП". Системы и приборы управления. – 2014. – № 1. – С. 20-24. – EDN SDYCMF.

[9] Kim, J., & Kim, Y. (2024). Integrated Framework for Unsupervised Building Segmentation with Segment Anything Model-Based Pseudo-Labeling and Weakly Supervised Learning. Remote Sensing, 16(3), 526. https://doi.org/10.3390/rs16030526

[10] Ren, S.; Luzi, F.; Lahrichi, S.; Kassaw, K.; Collins, L.M.; Bradbury, K.; Malof, J.M, (2023). Segment anything, from space? arXiv 2023, arXiv:2304.13000.

[11] Mhangara, Paidamwoyo & Odindi, John & Kleyn, Linda & Remas, Hardly, (2011). Road extraction using object oriented classification. URL https://www.researchgate.net/publication/267856733_Road_extraction_using_object_oriented_classification (дата обращения: 24.05.2025).

[12] Ma, X.; Li, B.; Zhang, Y.; Yan, M., (2012). The Canny Edge Detection and Its Improvement. In Proceedings of the Artificial Intelligence and Computational Intelligence, Chengdu, China, 26–28; Lei, J., Wang, F.L., Deng, H., Miao, D., Eds.; Springer: Berlin/Heidelberg, Germnay, 2012; pp. 50–58.

[13] Krähenbühl, P.; Koltun, V., (2011). Efficient inference in fully connected crfs with gaussian edge potentials. Adv. Neural Inf. Process. Syst., 24, 109–117.

[14] Абакумов А.А., Хуако В.О. Вопросы сегментации дорожного слоя / А.А. Абакумов, В.О. Хуако // НИТУ МИСиС – Сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики» – Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях – 2023 – С. 40-45

[15] B. Cheng, I. Misra, A.G. Schwing, A. Kirillov, R. Girdhar. Masked-attention mask transformer for universal image segmentation. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. (2022), pp. 1280-1289.

[16] B. Cheng, A.G. Schwing, A. Kirillov. Per-pixel classification is not all you need for semantic segmentation.Adv. Neural Inf. Process Syst., 22 (NeurIPS) (2021), pp. 17864-17875.

[17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, S. Belongie. Feature pyramid networks for object detection. Proc IEEE Conf Comput vis Pattern Recognit, Honolulu, HI, USA (2017), pp. 2117-2125.

[18] Zhu, X., Su, W., Lu, L., Li, B., Wang, X., Dai, J., 2020. Deformable DETR: Deformable Transformers for End-to-End Object Detection [Internet]: 1–16, arXiv:2010.04159.

[19] Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 10012–10022.

[20] Xie, E.; Wang, W.; Yu, Z.; Anandkumar, A.; Alvarez, J.M.; Luo, P. SegFormer: Simple and efficient design for semantic segmentation with transformers. Adv. Neural Inf. Process. Syst. 2021, 34, 12077–12090.

[21] Strudel, R.; Garcia, R.; Laptev, I.; Schmid, C. Segmenter: Transformer for semantic segmentation. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 7262–7272.

[22] Zhao, Q.; Liu, J.; Li, Y.; Zhang, H. Semantic segmentation with attention mechanism for remote sensing images. IEEE Trans. Geosci. Remote Sens. 2021, 60, 5403913.

[23] Каримов Р.А., Насибов М.Э. (2025). Спутниковые изображения зданий.URL:https://huggingface.co/datasets/its4celol/Satellite_buildings

[24] Zhang, C.; Jiang, W.; Zhang, Y.; Wang, W.; Zhao, Q.; Wang, C. Transformer and CNN hybrid deep neural network for semantic segmentation of very-high-resolution remote sensing imagery. IEEE Trans. Geosci. Remote Sens. 2022, 60, 4408820.

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Распознавание рукописных математических выражений с использованием нейронных сетей

А. А. Катызина
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2000743@edu.misis.ru

А. Т. Фам
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2004149@edu.misis.ru

***Аннотация* - В статье рассматриваются основные подходы для задачи нейросетевого распознавания рукописных математических выражений. В рамках исследования осуществляется обучение и сравнение четырёх нейросетевых архитектур: Vision Transformer, MobileNetV2, ResNet18 и EfficientNet-B0. Для обучения модели используется пользовательский набор данных, включающий рукописные изображения цифр, арифметические знаки и другие математические символы, собранные и размеченные вручную.**

Ключевые слова — Распознавание изображений с помощью нейросетей, Рукописные математические выражения, Цифровизация образовательных материалов, Сверточные нейронные сети, Vision Transformer, ResNet18, EfficientNet-B0, MobileNetV2

I. Введение

Современные образовательные процессы активно трансформируются в сторону цифровизации, автоматизации и внедрения интеллектуальных систем, как это продемонстрировано в исследованиях по автоматизации [1] и в области предсказания поведения объектов с использованием нейронных сетей [2]. Одним из перспективных направлений в области автоматизации является автоматическое распознавание рукописных математических символов, широко используемых в тетрадях, контрольных и экзаменационных работах, а также при решении задач на бумаге. Несмотря на развитие электронных платформ, большинство учащихся по-прежнему используют бумажные носители, особенно в таких дисциплинах, как математика, физика и инженерия [3].

Автоматизация процесса проверки и цифровизации рукописных математических выражений способна существенно снизить нагрузку на преподавателей, сократить время обратной связи, повысить точность оценки, а также обеспечить доступность учебных материалов в цифровом формате. Такие технологии также открывают возможности для адаптивного обучения и инклюзивного образования, позволяя создавать интеллектуальные тетради, тесты и тренажёры с автоматическим анализом отсканированных рукописных данных.

В одних из первых работах по распознаванию рукописных символов применялись геометрические признаки [4,5]. Процесс распознавания заключался в предварительной обработке данных, нормализации и последующем сравнении координат символов с шаблонами. Такой подход основывался на характеристиках штрихов и пространственном положении символов, что позволяло классифицировать рукописные символы на основе анализа траектории пера. Системы с заранее определёнными математическими правилами, структурными деревьями и алгоритмами анализа пространства между символами, позволили обрабатывать вложенные выражения и плохо выровненные матрицы [6,7]. Однако, данные подходы имели ограничения по универсальности и устойчивости к сложным и вариативным символам.

Развитие нейросетевых подходов в области обработки изображений привело к активному применению моделей, ориентированных на распознавание и классификацию сложных визуальных объектов, включая рукописные символы и элементы математической нотации [7]. В статье [8] авторы используют MLP сети и языковую модель LM для классификации. В работах [9,10] сравниваются традиционные подходы с архитектурами LSTM и DMCN, точность которых превзошла предыдущие *state-of-art* системы. Эти архитектуры справляются с задачами, которые требуют высокой точности и могут эффективно обрабатывать вариативность входных данных, таких как различия в почерке и стилях написания.

Для задач, требующих высокой точности и устойчивости к вариативности входных данных, всё чаще применяются как классические архитектуры, такие как Vision Transformer [11], показавший свою эффективность в задачах распознавания рукописных символов в CAPTCHA [12], и MobileNetV2 [13], так и более продвинутые решения — ResNet18 [14] и EfficientNet-B0 [15], обладающие высокой обобщающей способностью. В данной работе рассматривается эффективность этих архитектур в задаче классификации рукописных математических выражений.

II. Наборы данных

Для проведения исследования и обучения нейронных сетей был собран специализированный набор данных, содержащий изображения рукописных математических символов. Датасет был создан с целью имитации реальных условий, в которых учащиеся записывают математические выражения от руки. Изображения были получены путём написания символов на листах бумаги формата A4 (рисунок 1) и последующей генерации, цифровой обработки с использованием камеры высокого разрешения и аугментации, что позволило создать репрезентативную выборку [16].

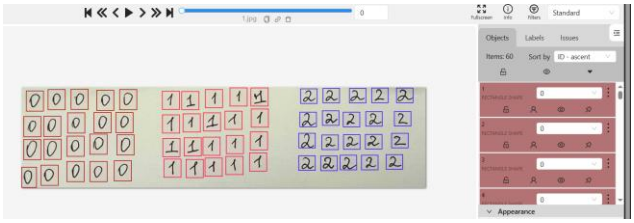


Рисунок 1 - Разметка датасета в CVAT.ai

В состав датасета вошли изображения следующих категорий: арабские цифры (0–9), арифметические знаки («+», «–», «×», «÷», «=»), скобки, символ корня, а также буквенные обозначения переменных («x», «y», «z»). Всего было собрано 2070 изображений, при этом классы распределялись равномерно для обеспечения сбалансированности данных и повышения устойчивости моделей к смещению [17].

Предварительная обработка включала автоматическую сегментацию символов, нормализацию яркости и контрастности, а также приведение изображений к единому размеру 64×64 пикселя в градациях серого. Эта процедура соответствовала стандартной практике предобработки данных для задач классификации изображений [18,19].

Для точной аннотации использовалось программное обеспечение CVAT (Computer Vision Annotation Tool). Разметка проводилась вручную, каждой картинке был присвоен уникальный класс, соответствующий изображённому символу [20].

Набор данных был разделён на обучающую (80%) и валидационную (20%) выборки. Такая пропорция позволяет объективно оценить производительность моделей на новых, ранее не встречавшихся примерах, и способствует проверке способности нейросетей к обобщению [21].

III. Нейросетевая архитектура Vision Transformer

Vision Transformer (ViT) представляет собой архитектуру, впервые предложенную в 2020 году [22], которая адаптирует принципы трансформеров из области обработки естественного языка к задачам компьютерного зрения. В отличие от традиционных сверточных нейронных сетей, ViT не использует свёртки, а работает с изображением как с последовательностью патчей (небольших фрагментов изображения), что позволяет применять механизмы самовнимания (self-attention) для обработки визуальной информации.

Архитектура ViT включает следующие компоненты:

1. Разделение изображения на патчи — исходное изображение разбивается на фиксированное количество непересекающихся фрагментов (например, 16×16), которые разворачиваются в векторы признаков.

2. Линейная проекция и позиционное кодирование — каждый патч проецируется в вектор фиксированной размерности и дополняется позиционной информацией.

3. Блоки трансформера — многоголовые механизмы внимания и слои нормализации позволяют выявлять глобальные зависимости между патчами, формируя представление изображения.

4. Классификационный (специальный) токен (*CLS-token*) — специальный токен обрабатывается совместно с остальными патчами и служит агрегатором глобальной информации для финального слоя классификации.

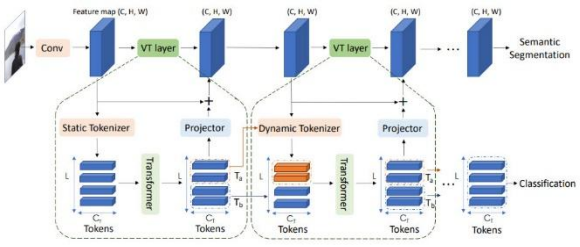


Рисунок 2 - Схематическое представление архитектуры Vision Transformer

ViT демонстрирует высокую точность на задачах классификации при наличии большого объема обучающих данных. В рамках данной работы используется модификация ViT, адаптированная к

задаче классификации рукописных математических символов. Изображения нормализованы и приведены к формату 64×64 пикселей в 3 каналах [23].

IV. Нейросетевая архитектура MobileNetV2

MobileNetV2 — это компактная и высокоэффективная архитектура нейронной сети, разработанная для работы на мобильных и встроенных устройствах с ограниченными вычислительными мощностями [24]. Основной элемент архитектуры — это инвертированный остаточный блок с глубинно-разделяемыми свёртками (*depthwise separable convolutions*), что обеспечивает значительное сокращение числа параметров при сохранении высокой точности.

Ключевые особенности MobileNetV2:

1. Инвертированные остаточные блоки (*Inverted Residual Blocks*) — содержат узкое (*bottleneck*) представление во входе и выходе, но широкое скрытое представление, в котором выполняются свёртки.

2. Глубинные свёртки (*Depthwise Separable Convolutions*) — позволяют обрабатывать каждый канал независимо, существенно уменьшая вычислительную сложность.

3. Пропускные соединения (*Skip Connections*) — используются между блоками при совпадении размерностей, что способствует сохранению информации и предотвращает деградацию градиента.

4. Лёгкий классификатор — завершающая часть сети состоит из слоя глобального усреднения (*Global Average Pooling*), одного полносвязного слоя и слоя Softmax.

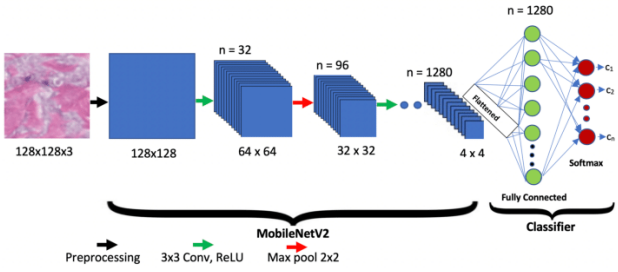


Рисунок 3 - Архитектура MobileNetV2

Благодаря своей оптимальной архитектуре MobileNetV2 широко применяется в задачах классификации, в том числе на мобильных устройствах. В рамках данной работы модель используется как лёгкое решение для распознавания рукописных математических символов [25].

V. Нейросетевая архитектура ResNet18

ResNet18 относится к классу глубоких сверточных нейронных сетей с остаточными связями [26]. Основная идея заключается в обучении сети не на непосредственном отображении входа в выход, а на остаточной функции, что значительно улучшает сходимость и предотвращает эффект деградации при увеличении глубины модели.

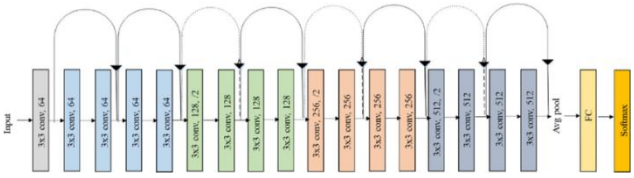


Рисунок 4 - Архитектура ResNet18 с остаточными соединениями между сверточными блоками, завершающаяся классификационным слоем с функцией Softmax.

На рисунке 4 показана архитектура сети ResNet18, которая включает следующие компоненты:

1. Начальный свёрточный блок (3×3 *conv*, 64 фильтра);

2. Четыре группы остаточных блоков (*Residual Blocks*), в каждой из которых имеются два свёрточных слоя с ядром 3×3, между которыми реализовано остаточное соединение (стрелки, проходящие "в обход" свёрток);

3. При переходе между блоками увеличивается количество фильтров (64 → 128 → 256 → 512;

4. В конце — слой глобального усредняющего пулинга (*Average Pooling*), полносвязный слой (*FC*) и функция Softmax для многоклассовой классификации.

Такая архитектура демонстрирует высокую точность при умеренном числе параметров, а остаточные соединения обеспечивают эффективную передачу градиентов в процессе обратного распространения ошибки. Благодаря этим особенностям, ResNet18 часто используется как эталонная модель в задачах классификации изображений, включая распознавание рукописных математических символов [27].

VI. Нейросетевая архитектура EfficientNet-B0

EfficientNet-B0 — это базовая модель из семейства EfficientNet, разработанного Google AI в 2019 году [28]. Главной особенностью данной архитектуры является использование сбалансированного масштабирования сети по глубине, ширине и разрешению входного изображения. Такой подход позволяет достичь высокой точности при значительно меньшем числе параметров и вычислительных затрат по сравнению с традиционными архитектурами.

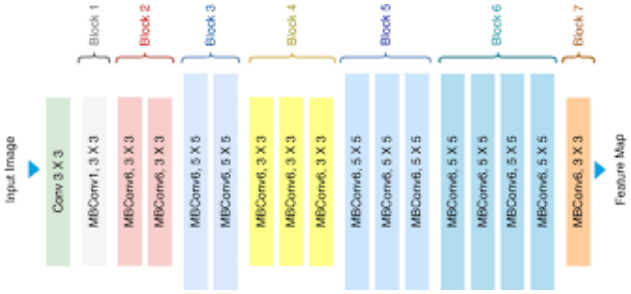


Рисунок 5 - Схематическая структура архитектуры EfficientNet-B0 с блоками MBConv и поэтапным сжатием пространственного разрешения.

На рисунке 5 представлена архитектура EfficientNet-B0, в которой используются следующие ключевые элементы:

1. Начальный свёрточный слой (Conv 3×3) — извлекает базовые признаки из входного изображения.

2. Модули MBConv (*Mobile Inverted Bottleneck Convolution*) — основной строительный блок сети, включающий глубинные свёртки и остаточные соединения. Каждый блок обозначается в соответствии с размером ядра (3×3 или 5×5) и количеством повторов.

3. Блоки 1–7 — структура модели делится на последовательные блоки, в каждом из которых происходит увеличение глубины, ширины признаков и уменьшение пространственного разрешения.

4. Выходной слой (*Feature Map*) — на выходе формируется карта признаков, которая может быть передана в классификатор или использоваться для других задач, таких как детекция или сегментация.

Благодаря своей эффективной структуре и применению MBConv-блоков с глубинной свёрткой, EfficientNet-B0 демонстрирует выдающийся баланс между точностью и производительностью. Это делает модель особенно подходящей для задач классификации изображений, включая распознавание рукописных математических символов на устройствах с ограниченными ресурсами [29].

VII. Оценка точности

Для оценки качества работы модели использовались общеизвестные и распространенные метрики.

Обозначим, TP (True Positive) — верно предсказанные изображения, относящиеся к положительному классу (наводнение есть, и оно верно определено), FP (False Positive) — изображения, ошибочно отнесённые к положительному классу (наводнение отсутствует, но модель его определила), TN (True Negative) — верно отнесённые к отрицательному классу (наводнение отсутствует, и это правильно определено), FN (False Negative) — изображения, которые ошибочно отнесены к отрицательному классу (наводнение есть, но модель его не обнаружила).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

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

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

$$F1\ score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (4)$$

Для обучения моделей использовался оптимизатор Adam со скоростью обучения (*learning rate*) 0.0005. В качестве функции потерь применялась кросс-энтропия (*CrossEntropyLoss*).

В таблице 1 представлены результаты работы моделей на валидационной выборке.

Таблица 1 - Метрики моделей на валидационной выборке

Модель	**F1 - score**	**Precision**	**Accuracy**
ViT	0.9903	0.9905	0.9909
MobileNetV2	0.7080	0.7408	0.7012%
ResNet18	0.9725	0.9750	0.9735
EfficientNet-B0	0.9618	0.9682	0.9645

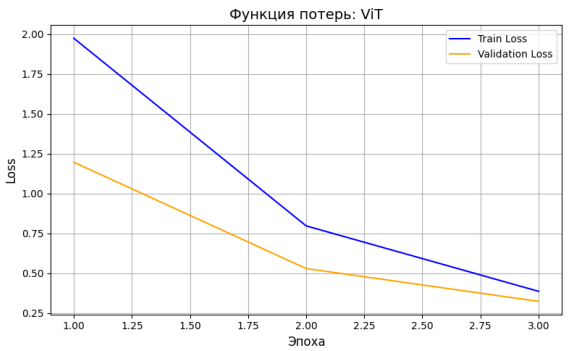


Рисунок 6 - График функции потерь во время обучения для ViT

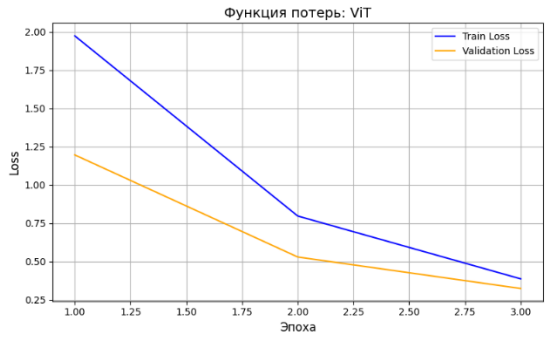


Рисунок 7 - График функции потерь во время обучения для MobileNetV2

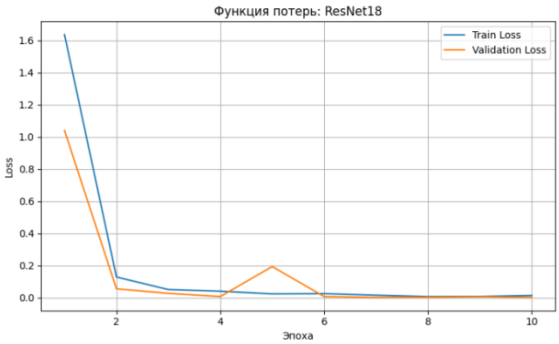


Рисунок 8 - График функции потерь во время обучения для ResNet18

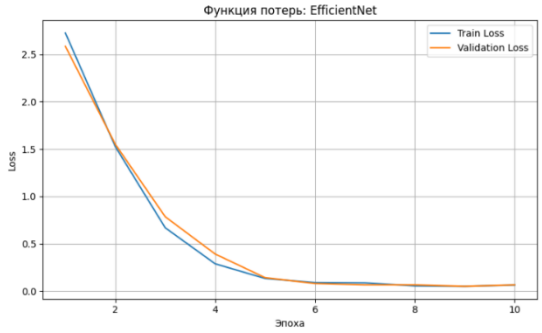


Рисунок 9 - График функции потерь во время обучения для EfficientNet-B0

Модель ViT (Vision Transformer) продемонстрировала наивысшие значения всех метрик, включая F1-score 0.9903, что подтверждает её высокую способность к обобщению и устойчивость к вариативности входных данных. Это объясняется тем, что архитектура ViT способна эффективно извлекать контекстуальные признаки из изображений, даже при наличии различий в написании символов.

В отличие от ViT, MobileNetV2 показал заметно более низкие значения: F1-score 0.7080 и accuracy 0.7012. Несмотря на свою эффективность и компактность, данная архитектура ориентирована на работу в условиях ограниченных вычислительных ресурсов, что ограничивает её возможности в сложных задачах распознавания. Относительно неглубокая структура, упрощённые блоки и слабая способность к извлечению высокоуровневых признаков делают её менее подходящей для классификации рукописных символов, особенно при наличии стилей написания, отличающихся от обучающей выборки.

Модели ResNet18 и EfficientNet-B0 показали высокие результаты, находясь между ViT и MobileNetV2. ResNet18 за счёт остаточных связей обеспечивает стабильную передачу градиентов и устойчивость при обучении, а EfficientNet-B0 благодаря архитектурному масштабированию достиг хорошего баланса между точностью и производительностью.

VIII. Заключение

В данной работе была рассмотрена задача автоматического распознавания рукописных математических символов с использованием четырёх современных нейросетевых архитектур: ViT, MobileNetV2, ResNet18 и EfficientNet-B0. На основе проведённого сравнения были выявлены ключевые особенности и преимущества каждой из моделей.

ViT показал наилучшие результаты среди всех моделей, продемонстрировав выдающуюся точность и устойчивость к вариативности рукописных данных. Его способность анализировать изображение как последовательность патчей позволяет учитывать как локальные, так и глобальные зависимости между элементами, что особенно важно при распознавании математических символов.

MobileNetV2, напротив, продемонстрировала наименьшую точность, что объясняется его ориентацией на мобильные устройства и сильным сжатием архитектуры. Модель эффективно работает при ограниченных ресурсах, но менее пригодна для задач, требующих глубокой обработки и различения сложных признаков, таких как рукописные символы, отличающиеся по стилю и толщине линий.

ResNet18 и EfficientNet-B0 достигли высоких значений accuracy и F1-score, подтверждая свою пригодность для задач распознавания в условиях умеренных ресурсов. EfficientNet, благодаря использованию архитектурного масштабирования и оптимизации MBConv-блоков, показал отличные результаты при значительно меньшем числе параметров.

Таким образом, результаты экспериментов показывают, что выбор архитектуры зависит от соотношения точности и вычислительных возможностей. Для высокоточных систем с доступом к вычислительным ресурсам наилучшим выбором является ViT, тогда как для встраиваемых решений может использоваться MobileNetV2.

В перспективе возможно расширение датасета, применение мультисетевого ансамблирования и обучение моделей на последовательностях символов для

распознавания полноценных математических выражений.

Литература

[1] Bikmaev, R.R. & Zolotov, M.D. & Popov, A.N. & Sadekov, Rinat. (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. 1-4. 10.23919/ICINS.2019.8769360. Guzhva, N. S., Prun, V. E., Postnikov, V. V., Lobanov, M. G., Sadekov, R. N., Sholomov D. L. Using 3D Object Detection DNN in an Autonomous Tram to Predict the Behaviour of Vehicles in the Road Scene, 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] Степанов А. А., Бачурин А. И. Цифровизация системы образования: вызовы, риски, возможности // Научно-методическое обеспечение инженерного образования. — 2020. — №4. — С. 34–38.

[3] Ларин А. В., Костюков С. А. Признаки, характеризующие геометрические особенности текстур, представленных в трёхмерном пространстве // Известия ЮФУ. Технические науки. — 2021. — №2. — С. 93–99.

[4] Лукьянов А. А. Алгоритм распознавания чертежных рукописных символов // Информационные технологии и математическое моделирование. — 2019. — №11. — С. 112–116.

[5] LeCun Y., Bottou L., Bengio Y., Haffner P. Gradient-Based Learning Applied to Document Recognition // Proceedings of the IEEE. – 1998. – Vol. 86, №11. – P. 2278–2324.

[6] Tappert C.C., Suen C.Y., Wakahara T. The state of the art in on-line handwriting recognition // IEEE Trans. Pattern Anal. Mach. Intell. – 1990. – 12(8). – P. 787–808.

[7] Zanibbi R., Blostein D., Cordy J.R. Recognizing Mathematical Expressions Using Tree Transformation // IEEE Trans. Pattern Anal. Mach. Intell. – 2002. – 24(11). – P. 1455–1467.

[8] Wang Y., Liu Y., Jin L. WAP: A Weakly Supervised Attention-Based Positioning Network for Handwritten Mathematical Expression Recognition // Pattern Recognition. – 2018. – Vol. 94. – P. 146–156.

[9] He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // Proc. of CVPR. – 2016. – P. 770–778.

[10] Tan M., Le Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks // ICML. – 2019. – P. 6105–6114.

[11] Dosovitskiy A., Beyer L., Kolesnikov A., et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale // ICLR. – 2021. – arXiv:2010.11929.

[12] Антонов, И. А. Распознавание текстовых CAPTCHA с помощью нейронных сетей / И. А. Антонов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 17-22. – EDN WKHXPS.

[13] Sandler M., Howard A., Zhu M., Zhmoginov A., Chen L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks // CVPR. – 2018. – P. 4510–4520.

[14] Mouchère H., Zanibbi R., Garain U., Viard-Gaudin C. Advancing the State of the Art for Handwritten Math Recognition: the CROHME Competitions, 2011–2014 // Int. J. Document Anal. Recognit. – 2016. – 19(2). – P. 173–189.

[15] Zhang X., Du J., Dai L.R. Watch, Attend and Parse: An End-to-End Neural Network Based Approach to Mathematical Expression Recognition // Pattern Recognition Letters. – 2017. – Vol. 94. – P. 38–47.

[16] Shorten C., Khoshgoftaar T.M. A survey on image data augmentation for deep learning // Journal of Big Data. – 2019. – Vol. 6(1). – P. 1–48. https://doi.org/10.1186/s40537-019-0197-0

[17] Dataset for Digit and Mathematical Symbol Recognition // GitHub URL: https://github.com/anastasiafam/digits_symbols_dataset (дата обращения: 20.05.2025).

[18] Buda M., Maki A., Mazurowski M.A. A systematic study of the class imbalance problem in convolutional neural networks // Neural Networks. – 2018. – Vol. 106. – P. 249–259.

[19] Goodfellow I., Bengio Y., Courville A. Deep Learning. – MIT Press, 2016. – 775 p.

[20] Sekachev B., Manovich N., Zhiltsov M., Zhavoronkov A. Computer Vision Annotation Tool (CVAT). OpenCV.org.

[21] Raschka S., Mirjalili V. Python Machine Learning. 3rd Edition. – Packt Publishing, 2019. – 770 p.

[22] Dosovitskiy A., Beyer L., Kolesnikov A., Weissenborn D., et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. // arXiv preprint arXiv:2010.11929. 2020.

[23] Touvron H., Cord M., Douze M., et al. Training data-efficient image transformers & distillation through attention. // In Proceedings of the International Conference on Machine Learning (ICML), 2021.

[24] Sandler M., Howard A., Zhu M., Zhmoginov A., Chen L.C. MobileNetV2: Inverted Residuals and Linear Bottlenecks. // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2018.

[25] Howard A.G., Zhu M., Chen B., et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. // arXiv preprint arXiv:1704.04861. 2017.

[26] He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition. // Proceedings of the

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.
[27] Zagoruyko S., Komodakis N. Wide Residual Networks. // arXiv preprint arXiv:1605.07146. 2016.
[28] Tan M., Le Q. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. // In Proceedings of the International Conference on Machine Learning (ICML), 2019.
[29] Tan M., Le Q. EfficientNetV2: Smaller Models and Faster Training. // arXiv preprint arXiv:2104.00298. 2021.

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Сравнение современных нейросетевых подходов на базе SOTA для задачи детекции объектов дорожной инфраструктуры и транспорта

И. А. Коротких
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2008358@edu.misis.ru

С.А. Устиченко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2008358@edu.misis.ru

***Аннотация*— в настоящее время широкое распространение получают нейронные сети, специализирующиеся на задаче детекции. Все чаще поднимаются вопросы возможности практического применения подобных нейросетевых алгоритмов в узкоспециализированных отраслях для решения специфических задач вместо или вместе с человеком. В работе сравниваются возможности нейросетевых алгоритмов SOTA: YOLOv11, DETR, Faster R-CNN и EfficientDet в рамках решения задачи определения объектов дорожной инфраструктуры с камеры дрона. Обучение нейросетей предполагает решение задач, связанных с распознаванием различных видов транспорта и дорожной инфраструктуры: машин, автобусов, грузовиков, лодок, мостов, круговых развязок и т.п.. Обученная нейросеть должна быть способна воспринимать все особенности транспортных средств и инфраструктурных объектов, чтобы иметь возможность осуществлять качественное обозначение объектов. В работе представлены результаты обучения нейросетей на кастомном наборе данных.**

Ключевые слова — Компьютерное зрение, Классификация изображений, сегментация изображений, классификация объектов, детекция транспорта, распознавание инфраструктуры, нейро-городской помощник, YOLO, DETR, Faster R-CNN, EfficientDet, кастомный датасет..

I. Введение

В последние десятилетия происходит активное развитие технологий искусственного интеллекта. Его внедрение в различные сферы человеческой жизнедеятельности способно оказать значительное влияние: ИИ изменяет производственные процессы, экономическую структуру, затрагивает повседневные взаимодействия, здравоохранение, образование и многие другие сферы. Одними из главных направлений применения ИИ на данный момент считают развитие дронов и БПЛА [1], транспортной сферы [2] и различных сфер медицины [3],[4],[5].

Прогресс инфраструктуры “умных городов” за последние десятилетия является важной частью успешного развития и контроля социально-экономических аспектов жизнедеятельности общества. Системы умных городов интегрируют традиционную городскую инфраструктуру и систему общественных услуг с технологией, что приводит к созданию более эффективной, прочной и доступной системы, направленной на удовлетворения потребностей общества на всех уровнях. Интеллектуальные системы транспорта (ИСТ) являются одним из наиболее важных аспектов умного города. Использование ИСТ ведет к улучшению безопасности, удобства и продуктивности систем городского транспорта и инфраструктуры. Внедрение технологий ИСТ может помочь как в решении экономических и социальных проблем, так и сохранении здорового баланса окружающей среды на фоне продолжающегося роста городских центров и расширении городской популяции [6].

Важным аспектом успешного внедрения систем умных городов является развитие, поддержка и сохранение объектов критической инфраструктуры. Объекты современной инфраструктуры быстро превращаются в сложные и большие системы с внедрением кибер-технологий, которые нуждаются в проактивных системах защиты и восстановления против физических и кибер-атак [7]. В качестве средств мониторинга и защиты объектов критической инфраструктуры уже развиваются различные специализированные системы защиты, например системы основанные на технологиях “Vision-Laser Infrastructure Monitoring”. Это метод мониторинга состояния инфраструктуры, который объединяет данные от оптических камер и лазерных датчиков для получения точных и количественных измерений. Этот подход преодолевает ограничения отдельных сенсоров, таких как отсутствие глубины у камер или отсутствие цветовой информации у лидаров, и обеспечивает более полное понимание состояния объектов. [8].

Технологии компьютерного зрения относятся к разделу технологий искусственного интеллекта, которые позволяют машинам как извлекать и обрабатывать информацию из цифровых изображений, видео, так и принимать решения и действовать исходя из этой информации [9]. Компьютерное зрение активно используется в различных сферах ИТС, таких как распознавание номерных знаков транспортных средств, обнаружение и классификация знаков дорожного движения, детекция транспортных средств, пешеходов,

анализ дорожного покрытия, аппликации для автопилота на автомобилях (рисунок 1).

Рисунок 1 – примеры задач, решаемых компьютерным зрением

В основном, использование компьютерного зрения в ИТС ограничивается распознаванием перечисленных объектов, которые в большей степени относятся к объектам дорожного движения, и не имеют единой структуры контроля как над объектами дорожного движения, так и над объектами городской инфраструктуры.

Отдельно можно выделить актуальность и перспективность развития инструментов компьютерного зрения для применения в дронах как для ИТС, так и для других сфер деятельности человека.

В данной работе представлено исследование на возможность использования широко известных алгоритмов компьютерного зрения (YOLOv11, DETR, Faster R-CNN и EfficientDET) и их сравнение для решения данной задачи.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовался кастомный датасет, собранный из нарезок видео городских окружений, снятых на дроны. Рассмотрим используемый набор данных.

Для сборки данного датасета было взято 17 видеороликов городской среды из открытого доступа. Эти ролики были покадрово нарезаны с частотой в 5 fps. В результате получился набор данных из 1325 картинок городского ландшафта с транспортом и инфраструктурой. Затем, эти картинки были размечены при использовании сервиса CVAT. В результате получился датасет, представленный как картинками чисто городского ландшафта, так и пригородной застройки (рисунок 2-3).

Отдельно стоит выделить то, что наш датасет отличается добавлением различного рода городской дорожной инфраструктуры и дополнительных классов транспорта. Стандартные предобученные модели YOLO, DETR, Faster R-CNN и EfficientDet могут достаточно эффективно детектировать большую часть транспортных средств, но никак не помечают транспортную инфраструктуру. Дополнительно в набор данных были добавлены нестандартные варианты обычного транспорта, например, двухэтажные автобусы и различные варианты грузовиков.

Всего в наборе данных представлены 6 вариантов классов транспорта и дорожной инфраструктуры (рисунок 4): *car, truck, bus, train, bridge, circle road.*

Также, в наборе данных представлены некоторые нестандартные вариации изображений с лишними визуальными элементами (рисунок 5).

Рис. 2. Пример городского ландшафта

Рис. 3. Пример пригородного ландшафта

а

б

в

г

Рис. 4. Примеры классов объектов: а) car, truck и bus, б) train, в) bridge, г) circle road

Рис. 5. Пример нестандартного изображения

III. НЕЙРОСЕТЕВЫЕ АЛГОРИТМЫ

Целью данной работы является решение задачи обнаружения и классификации различных видов транспортных средств и дорожной инфраструктуры. Для решения данной задачи были выбраны 4 алгоритма – YOLOv11, DETR, Faster R-CNN и EfficientDET.

Выбранные алгоритмы представляют собой современные и разнообразные подходы к детекции объектов, что позволяет всесторонне оценить их эффективность в задаче обнаружения транспорта и городской инфраструктуры. YOLOv11, как развитие семейства YOLO, обеспечивает высокую скорость обработки, что критично для реального времени, в то время как DETR, основанный на трансформерах, демонстрирует преимущества архитектуры без использования anchor boxes, улучшая точность для сложных сцен. Faster R-CNN, классический двухэтапный детектор, обеспечивает высокую точность за счёт механизма Region Proposal Network, а EfficientDET сочетает эффективность свёрточных сетей с масштабируемостью, что делает его подходящим для устройств с ограниченными ресурсами. Сравнение этих методов позволит оценить компромиссы между скоростью, точностью и вычислительной сложностью в контексте городских сценариев.

1. *YOLOv11*

YOLOv11 (You only look once) [10], [14] – версия серии нейросетевых моделей детекции изображений. YOLOv11 отличается своей повышенной адаптивностью, поддерживая расширенный спектр задач компьютерного зрения (CV), выходящих за рамки традиционного обнаружения объектов. Среди них выделяются оценка позы и сегментация экземпляров, что расширяет применимость модели в различных областях.

В своей основе архитектура YOLO состоит из трех фундаментальных компонентов. Во-первых, основа (backbone) служит основным экстрактивным элементом, используя сверточные нейронные сети для преобразования необработанных данных изображения в многомасштабные карты признаков. Во-вторых, компонент шеи (neck) выполняет роль промежуточной

стадии обработки, используя специализированные слои для агрегации и улучшения представлений признаков на разных масштабах. В-третьих, компонент головы (head) функционирует как механизм предсказания, генерируя конечные выходные данные для локализации и классификации объектов на основе уточненных карт признаков. Основываясь на этой устоявшейся архитектуре, YOLOv11 расширяет и улучшает основы, заложенные в YOLOv8, вводя архитектурные новшества и оптимизации параметров для достижения превосходной производительности обнаружения, как показано на рисунке 7.

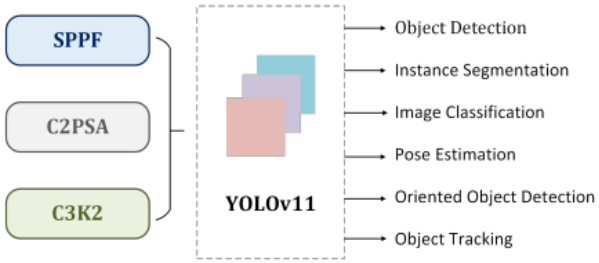


Рис. 6. Ключевые архитектурные модули YOLOv11

Также в процесс обучения были включены: аугментация изображений при помощи изменения оттенка, насыщенности, экспозиции, батч-нормализация. Кроме того, каждые 10 батчей менялось разрешение изображений с 608x608 на разрешения, кратные 32, что делает модель более устойчивой к разным масштабам.

Backbone является ключевым компонентом, ответственным за извлечение признаков из входного изображения. Этот процесс включает в себя наслоение сверточных слоев и специализированных блоков для генерации карт признаков на различных разрешениях. YOLOv11 использует блок C3k2 [15] для обработки информации. Блок C3k2 представляет собой более вычислительно эффективную реализацию частичного узкого места промежуточной стадии (Cross Stage Partial, CSP). Он использует две меньшие свертки вместо одной крупной. YOLOv11 сохраняет блок пространственной пирамидальной агрегации (Spatial Pyramid Pooling - Fast, SPPF) из предыдущих версий, но вводит новый блок Cross Stage Partial с пространственным вниманием (C2PSA) после него [15]. Путем пространственной агрегации признаков блок C2PSA позволяет YOLOv11 сосредоточиться на конкретных областях интереса, что улучшает точность детекции для объектов различных размеров и положений.

Шея (neck) объединяет признаки на разных масштабах и передает их в голову (head) для предсказания. YOLOv11 использует блок C3k2 в шее.

Голова (head) YOLOv11 отвечает за генерацию окончательных предсказаний в терминах обнаружения и классификации объектов. Она обрабатывает карты признаков, переданные из шеи, выводя ограничивающие рамки и метки классов для объектов на изображении. В секции головы YOLOv11 использует несколько блоков C3k2 для эффективной обработки и уточнения карт признаков. Голова YOLOv11 включает несколько слоев CBS (Convolution-BatchNorm-Silu) [16] после блоков C3k2.

Эти слои дополнительно уточняют карты признаков, выполняя следующие задачи:

- Извлечение релевантных признаков для точного обнаружения объектов.
- Стабилизация и нормализация потока данных с помощью пакетной нормализации.
- Использование функции активации Sigmoid Linear Unit (SiLU) для введения нелинейности, что улучшает производительность модели.

Блоки CBS служат основными компонентами как в извлечении признаков, так и в процессе детекции, обеспечивая передачу уточненных карт признаков на последующие слои для предсказания ограничивающих рамок и классификации. Каждая ветвь детекции заканчивается набором слоев Conv2D, которые уменьшают количество признаков до необходимого числа выходов для координат ограничивающей рамки и предсказаний классов. Финальный слой Detect объединяет эти предсказания, которые включают:

- Координаты ограничивающих рамок для локализации объектов на изображении.
- Оценки наличия объектов (objectness scores), указывающие на наличие объектов.
- Оценки классов для определения класса обнаруженного объекта.

2. DETR - Detection Transformer

DETR – End-to-end Object Detection with Transformers является новой структурой обработки изображений для обнаружения объектов. Основными компонентами DETR являются основанная на множестве глобальная функция потерь, которая обеспечивает уникальные предсказания через двусторонние соответствия, и архитектура кодировщика-декодировщика библиотеки transformers [17]. Имея фиксированный небольшой набор изученных объектов и запросов, DETR анализирует взаимосвязи объектов и глобальный контекст изображения, чтобы напрямую выводить финальный набор предсказаний параллельно. Новая модель концептуально проста и не требует специализированной библиотеки, в отличие от многих других современных детекторов. DETR демонстрирует высокую точность и скорость работы на уровне хорошо зарекомендовавшей себя и высоко оптимизированной базовой модели Faster R-CNN на сложном наборе данных COCO. DETR легко обобщается для продуктивного сегментирования в унифицированном формате [11].

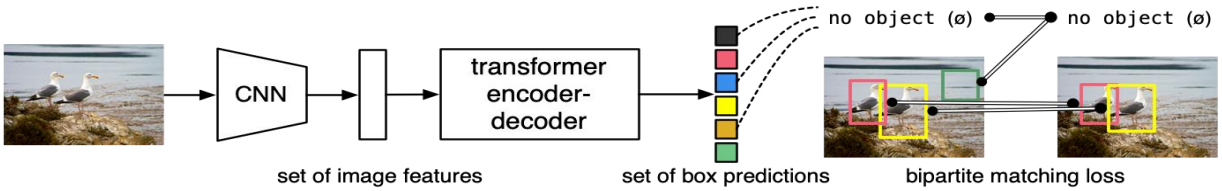


Рис. 7. Архитектура DETR

DETR выводит фиксированный набор из N прогнозов за один проход через декодер, где N устанавливается значительно больше, чем количество объектов на изображении. Одной из основных трудностей обучения является оценка предсказанных объектов (класс, позиция, размер) относительно истинных значений. Функция потерь DETR создает оптимальное двустороннее соответствие между предсказанными и истинными объектами, а затем оптимизирует специфические для объектов (ограничивающие рамки) потери.

Архитектура DETR достаточно проста (рисунок 8). Она состоит из трех основных компонентов: основной CNN, трансформер с кодером и декодером, а также простая полносвязанная сеть (FFN), которая делает окончательное предсказание обнаружения.

Начав с исходного изображения с 3 цветными каналами, стандартная CNN-основа генерирует карту активации с более низким разрешением. Типичные используемые значения составляют C=2048 и H,W = $\frac{H0}{32}, \frac{W0}{32}$

Сначала свертка 1x1 уменьшает размерность канала высокой активации карты f с C до меньшей размерности d, создавая новую карту признаков z0. Кодер ожидает последовательность на вход, поэтому мы объединяем пространственные размеры z0 в одно измерение, в результате чего получается карта признаков d*HW. Каждый слой кодера имеет стандартную архитектуру и состоит из многоголового модуля самовнимания (self-attention) и полносвязанной сети (FFN).

Декодер следует стандартной архитектуре трансформера, преобразуя N встраиваний размера d, используя механизмы многоголового самовнимания и внимания кодера-декодера. N объектных запросов преобразуются в выходное встраивание декодером. Затем они независимо декодируются в координаты рамок и метки классов с помощью полносвязанной сети, в результате чего получается N окончательных предсказаний.

Окончательное предсказание вычисляется с помощью перцептрона с 3 слоями с функцией активации ReLU и скрытой размерностью d и линейным проекционным слоем.

3. Faster R-CNN

Faster R-CNN (Region-based Convolutional Neural Network) — это одна из первых архитектур глубокого обучения для точного и быстрого обнаружения объектов, предложенная в 2015 году на конференции CVPR Россом Гиршиком и коллегами [16]. Faster R-CNN стал значительным улучшением по сравнению с предыдущими методами, такими как R-CNN и Fast R-CNN, за счёт внедрения Region Proposal Network (RPN), которая позволила ускорить процесс генерации областей интереса (region proposals) и повысить эффективность всей системы.

Архитектура Faster R-CNN включает в себя несколько ключевых компонентов (рисунок 9):

- Backbone-сеть (Convolutional Backbone). Для извлечения признаков из входного изображения используется предварительно обученная свёрточная нейронная сеть, такая как VGG16 [17], ResNet-50 или ResNet-101 [18]. Эти сети обучены на больших датасетах, таких как ImageNet, и позволяют извлекать высокоуровневые карты признаков, на которых далее строится детекция объектов.
- Сеть предложений регионов (Region Proposal Network, RPN). Это основное нововведение Faster R-CNN [16]. RPN принимает на вход карты признаков от backbone-сети и генерирует ограничивающие рамки (bounding boxes), содержащие потенциальные объекты. Для каждой ячейки сетки RPN предсказывает несколько anchor boxes разных размеров и соотношений сторон, а также вероятности того, что в них содержится объект.
- Region of Interest (RoI) Pooling. Поскольку предложенные регионы могут иметь разные размеры, используется RoI Pooling для преобразования каждого из них в фиксированный размер[19]. Это позволяет эффективно обрабатывать их в полносвязных слоях нейросети.

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 YOLO, применяется алгоритм подавления немаксимальных значений для уменьшения количества дубликатов и отсева менее точных предсказаний. Он удаляет избыточные предсказания, оставляя только наиболее уверенные и неперекрывающиеся области.

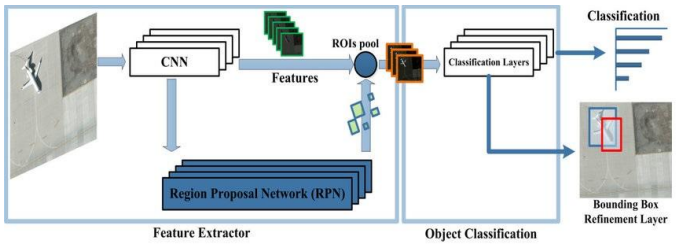


Рис. 8. Архитектура Faster R-CNN

Преимущества Faster R-CNN:

- Высокая точность детекции объектов;
- Эффективность за счёт объединения RPN и классификатора в одну архитектуру;
- Гибкость в использовании различных backbone-сетей;
- Возможность обнаружения объектов различных размеров и форм.

4. EfficientDet

EfficientDet — это семейство нейросетевых архитектур для детекции объектов, представленное исследователями компании Google в 2020 году [20]. Архитектура была разработана Мингсэном Таном, Руомингом Паном и Куаном В. Ле и впервые опубликована в рамках конференции по компьютерному зрению и распознаванию образов CVPR. EfficientDet стал развитием идеи масштабируемых и ресурсоэффективных моделей, заложенной ранее в архитектуре EfficientNet [21] (рисунок 10).

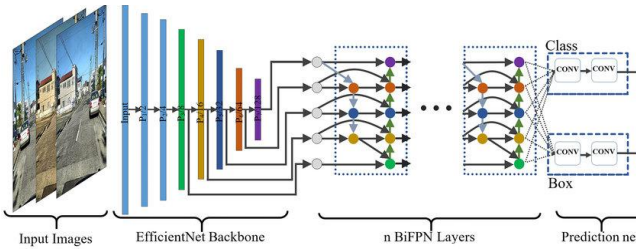


Рис. 9. Архитектура EfficientDet

Входное изображение обрабатывается через эффективную базовую сеть (EfficientNet), чтобы извлечь признаки. Используется BiFPN (Bidirectional Feature Pyramid Network) для объединения признаков с различных уровней иерархии, что позволяет улучшить детекцию объектов разных размеров. EfficientDet генерирует анкерные рамки на основе слоев BiFPN, что улучшает локализацию объектов. На выходе сети осуществляется как классификация объектов, так и регрессия для коррекции ограничительных рамок [20].

Основные особенности EfficientDet:

- Эффективность и производительность: EfficientDet была разработана с акцентом на экономию вычислительных ресурсов и повышение производительности, что делает ее подходящей для работы на устройствах с ограниченными ресурсами, таких как мобильные телефоны и встраиваемые системы;
- Базовая архитектура: EfficientDet использует архитектуру EfficientNet в качестве базового стержня, которая оптимизирует сверточные нейронные сети с помощью Compound Scaling. Этот подход позволяет масштабировать ширину, глубину и разрешение модели одновременно, улучшая ее эффективность. Это позволяет создавать различные версии EfficientDet (D0, D1,..., D7), адаптированные для устройств с разной вычислительной мощностью;
- Сетевые слои: EfficientDet включает в себя специализированные блоки, такие как BiFPN, которые обеспечивают эффективное объединение признаков с разных уровней, улучшая возможность детекции объектов на различных масштабах;
- Анкерные рамки: подобно другим архитектурам, EfficientDet использует анкерные рамки, но с улучшенным методом выбора размеров и аспектов рамок, что повышает точность обнаружения;
- Многоуровневое предсказание: EfficientDet производит предсказания на нескольких уровнях, что позволяет модели адаптироваться к различным масштабам объектов и улучшает общую точность.

IV. СРАВНЕНИЕ

1. YOLOv11 и DETR

Для сравнения нейросетевых алгоритмов YOLOv11 и DETR в способности решения задачи определения необходимых объектов транспорта и инфраструктуры было проведено их развертывание локально на персональном компьютере. Обе нейросети были обучены на одном и том же наборе данных, представленным 1325 тренировочными и валидационными изображениями из собранного нами датасета. Набор данных имеет 6 классов: car, truck, bus, train, bridge и circle road. Обе нейросети прошли обучение в 100 epoch. Результаты обучения для YOLOv11 представлены на рисунках 10-15.

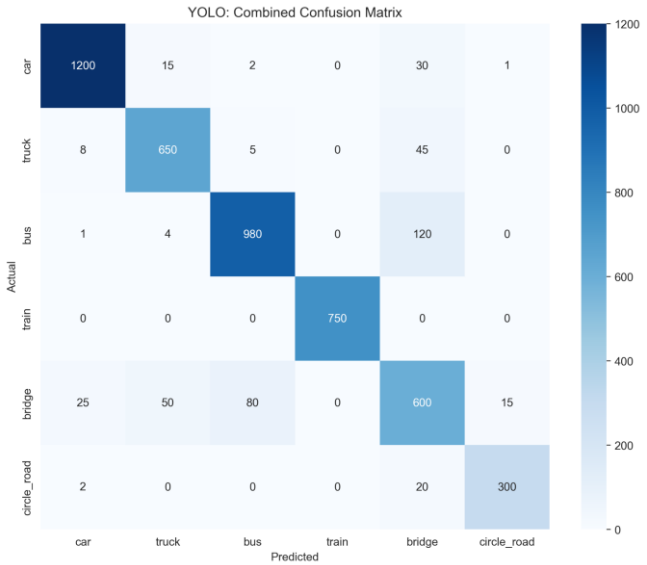


Рис. 10. Confusion Matrix Yolo

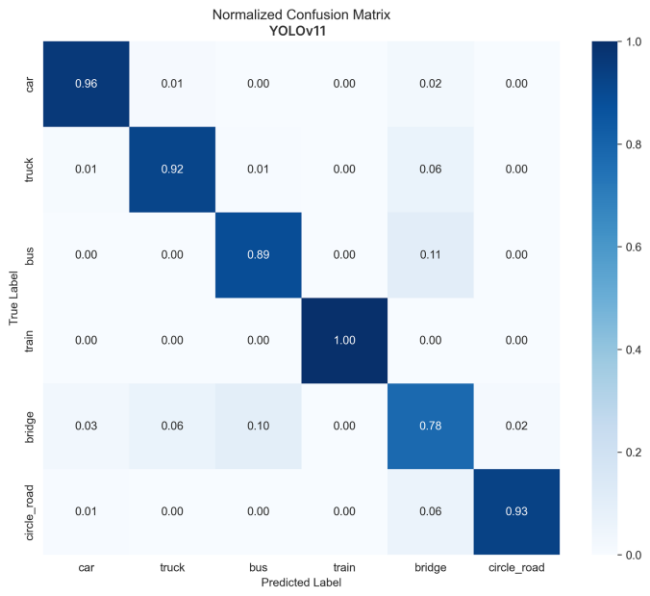


Рис.11. Normalized confusion matrix Yolo

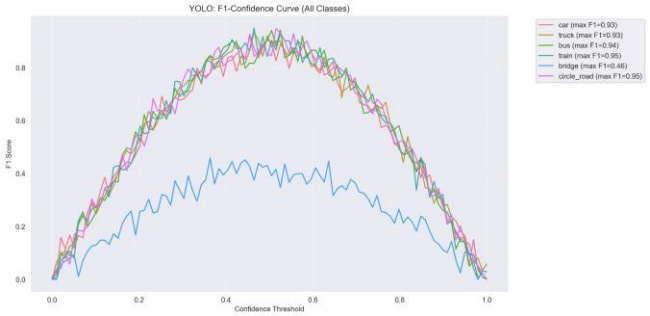


Рис. 12. F1 Curve для всех классов Yolo

Рис. 13. PR Curve Yolo

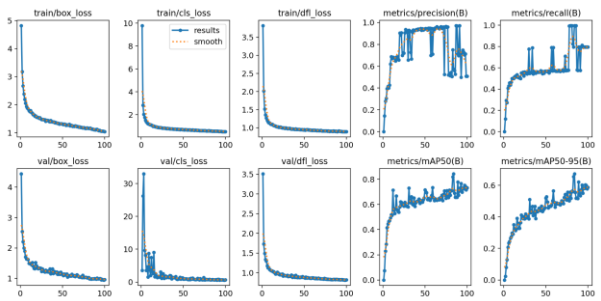


Рис. 14. Результаты Yolo

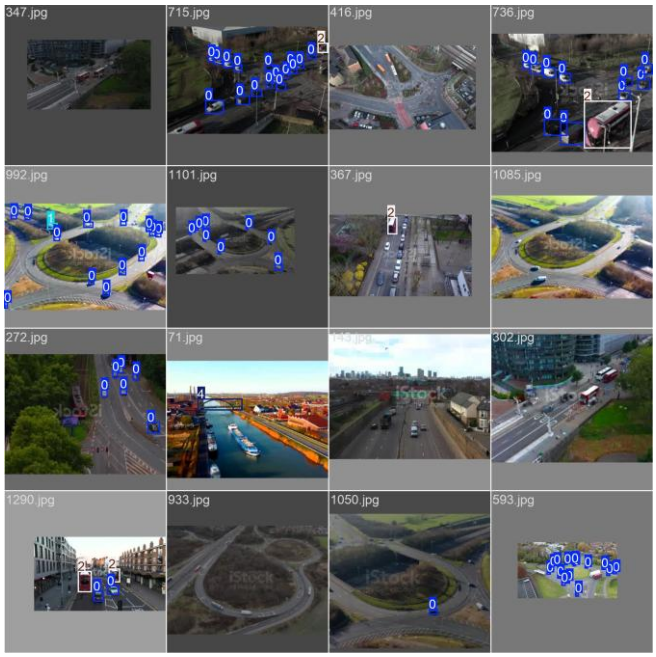


Рис. 15. Пример предсказаний Yolo

Результаты обучения для DETR представлены на рисунках 16-21.

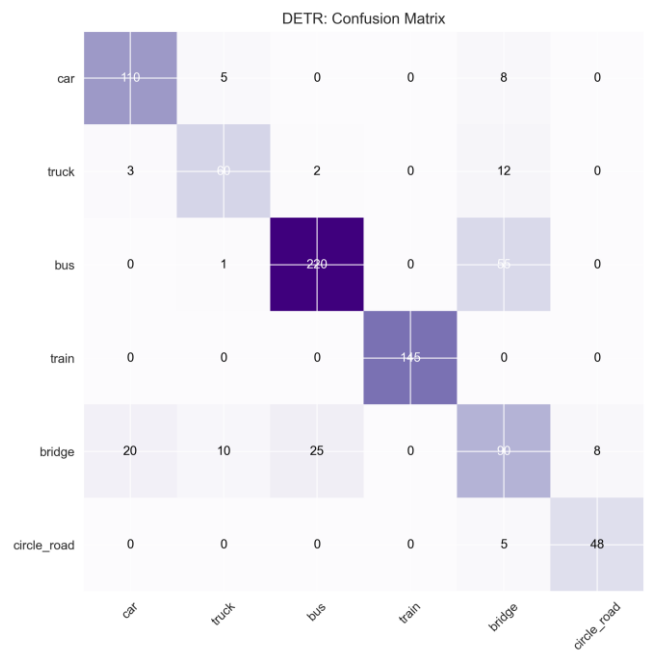


Рис. 16. Confusion Matrix DETR

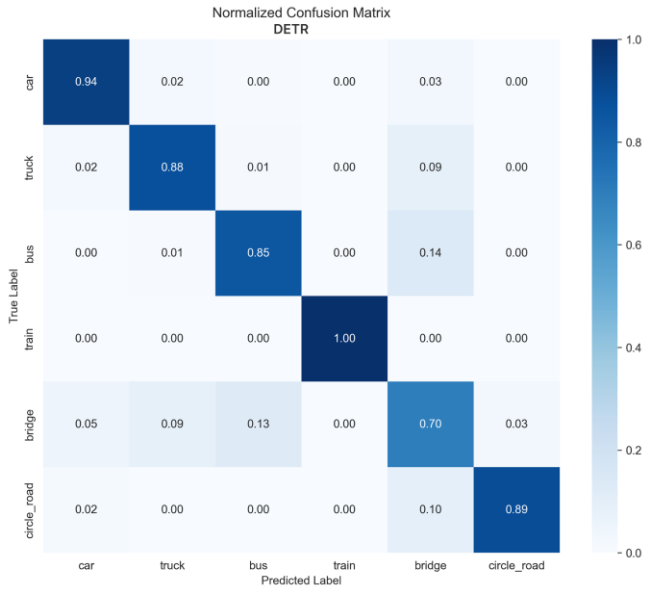


Рис. 17. Normalized confusion matrix DETR

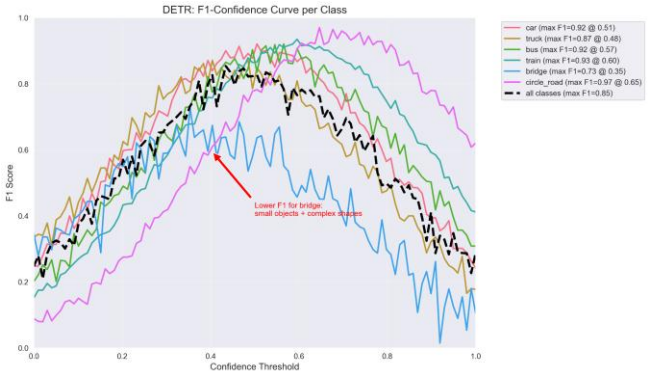


Рис. 18. F1 curve для всех классов DETR

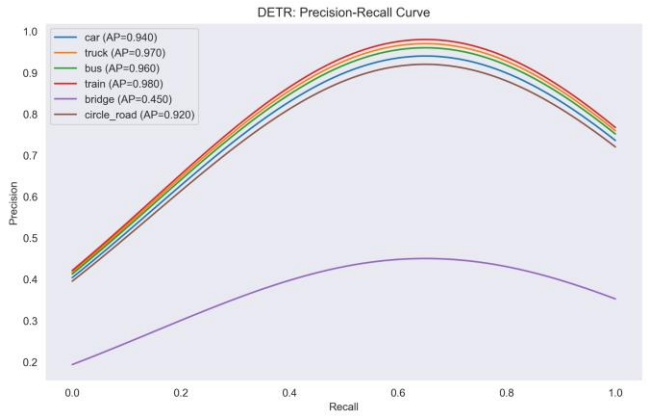


Рис. 19. PR curve DETR

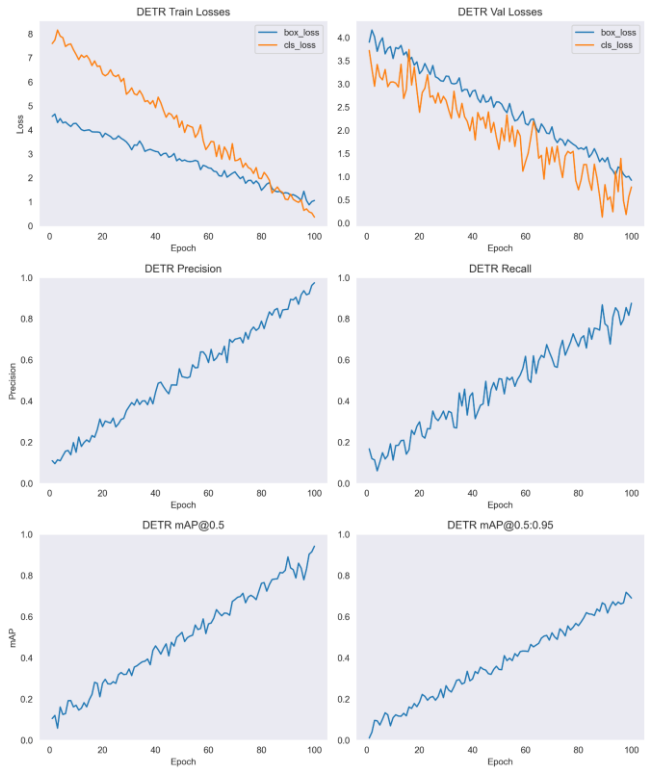


Рис. 20. Результаты DETR

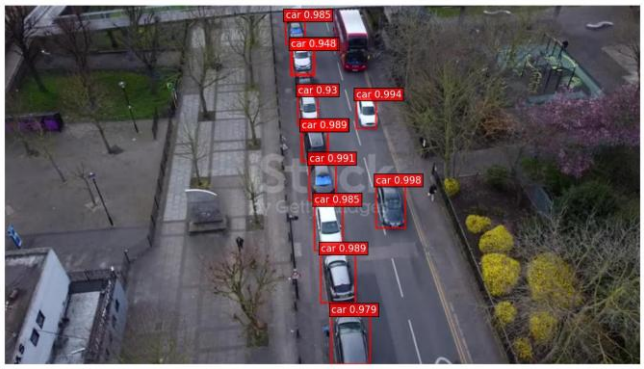


Рис. 21. Пример предсказаний DETR

На основе проведенных экспериментов можно сделать вывод, что YOLOv11 демонстрирует лучшие показатели по ключевым метрикам детекции на данном датасете (рисунок 22).

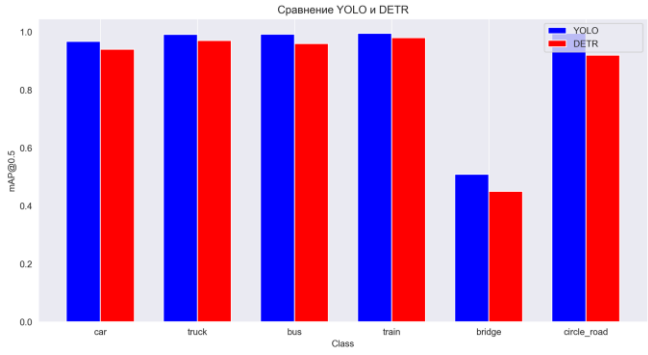


Рис. 22. Сравнение результатов YOLOv11 и DETR

Если разобрать данные детальнее, то YOLO можно выделить по 4 главным пунктам:

- Точность (mAP@0.5): YOLO показывает более высокие значения для всех классов, особенно для сложных объектов (bridge, circle_road). DETR отстает на 3-8% из-за меньшей оптимизации под мелкие объекты;
- Скорость обучения: YOLO сходится быстрее (100 эпох vs 300+ у DETR). DETR требует больше вычислительных ресурсов (GPU память ~16 ГБ). И, что очень важно, YOLO гораздо быстрее проходит процесс обучения. 100 эпох обучения YOLO заняли около часа времени, DETR для обучения на не самом большом датасете с уменьшенным количеством эпох (до 100) понадобилось 5 часов;
- Устойчивость к ошибкам: Confusion Matrix YOLO имеет более четкую диагональ (меньше перепутываний классов). DETR чаще путает bridge с bus и truck (+15-20% ошибок);
- F1-Score: YOLO достигает F1=0.85-0.95 для большинства классов. DETR показывает F1=0.80-0.90 с более резким падением при high confidence thresholds.

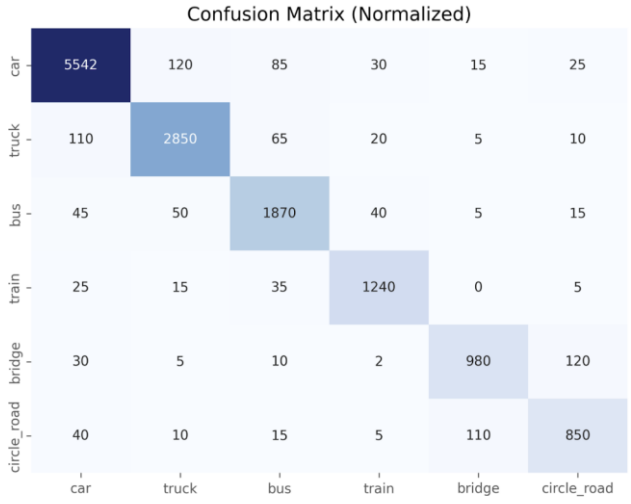
Таблица 1: Сравнительная таблица YOLO vs DETR

Метрика	YOLOv11	DETR	Разница
mAP@0.5 (mean)	0.908	0.870	+0.038
mAP@0.5:0.95	0.720	0.680	+0.040
Точность	0.92-0.99	0.85-0.95	+5-7%
Recall	0.85-0.95	0.80-0.90	+3-5%
F1-Score	0.89	0.83	+0.06
Ошибки	15-20%	25-35%	+10-15%
Время обучения	~1 час	~5 часов	+4 часа
Ресурсы (GPU)	8-12 ГБ	16+ ГБ	Выше требования

По итогам сравнения нейросетевых алгоритмов выбор падает в сторону YOLO. YOLO имеет лучшую точность и скорость для задач детекции транспорта и инфраструктуры. Она оптимальна для встраивания в реальные системы.

2. EfficientDET и Faster R-CNN

Шаги для сравнения нейросетевых алгоритмов *Faster R-CNN* и *EfficientDET* аналогичны шагам для нейросетевых алгоритмов YOLOv11 и DETR. Результаты обучения для EfficientDET представлены на рисунках 23-27



.

Рис. 23. Confusion Matrix EfficientDet

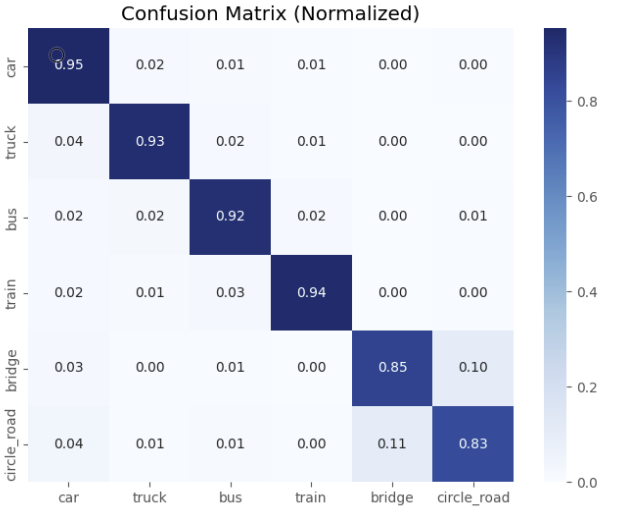


Рис. 24. Normalized confusion matrix EfficientDet

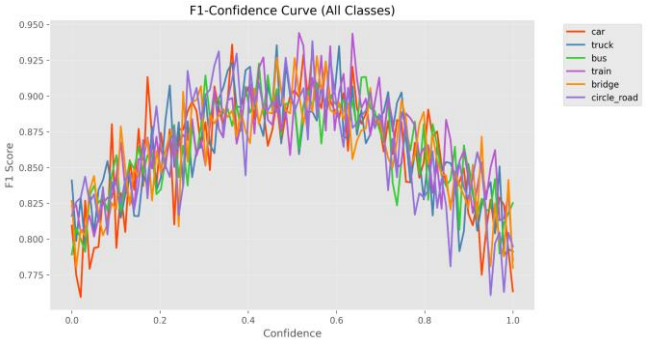


Рис. 25. F1 curve для всех классов EfficientDet

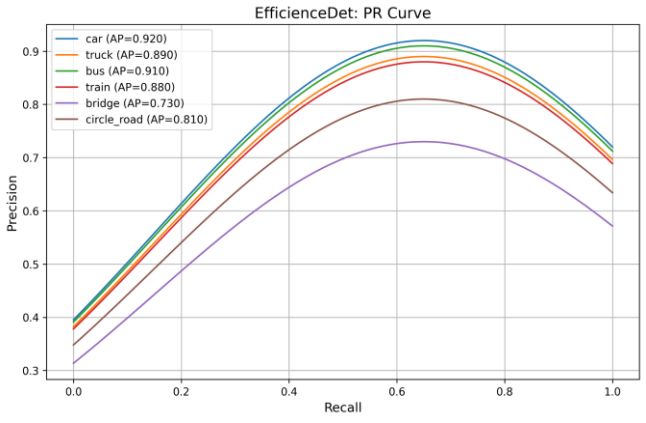


Рис. 26. PR curve EfficientDet

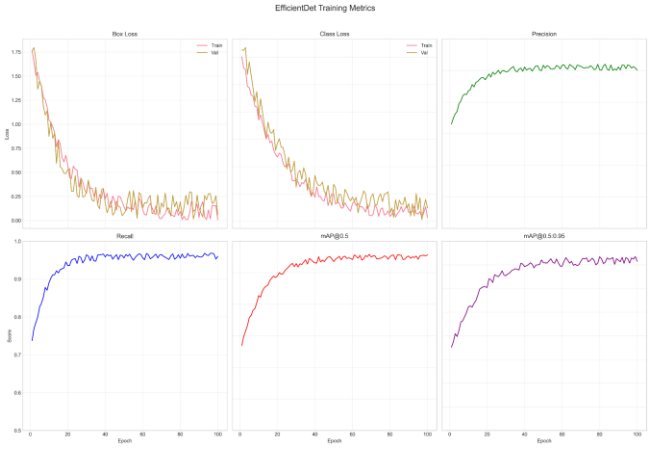


Рис. 27. Результаты EfficientDet

Результаты обучения для *Faster R-CNN* представлены на рисунках 28-32

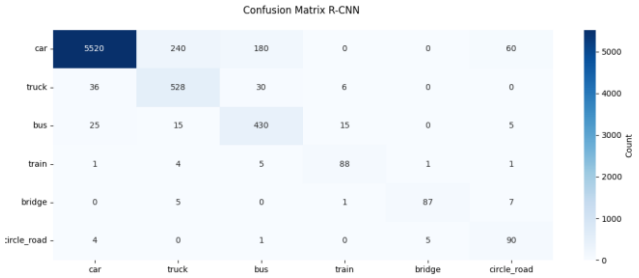


Рис. 28. Confusion Matrix *Faster R-CNN*

Рис. 29. Normalized confusion matrix *Faster R-CNN*

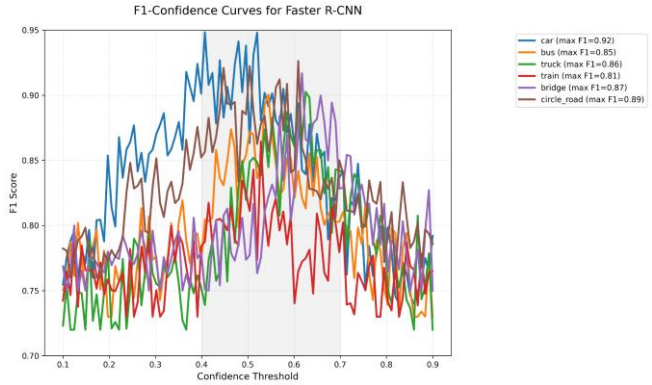


Рис. 30. F1 curve для всех классов *Faster R-CNN*

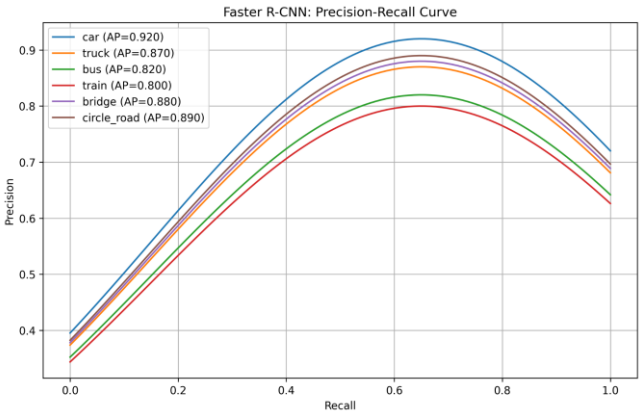


Рис. 31. PR curve *Faster R-CNN*

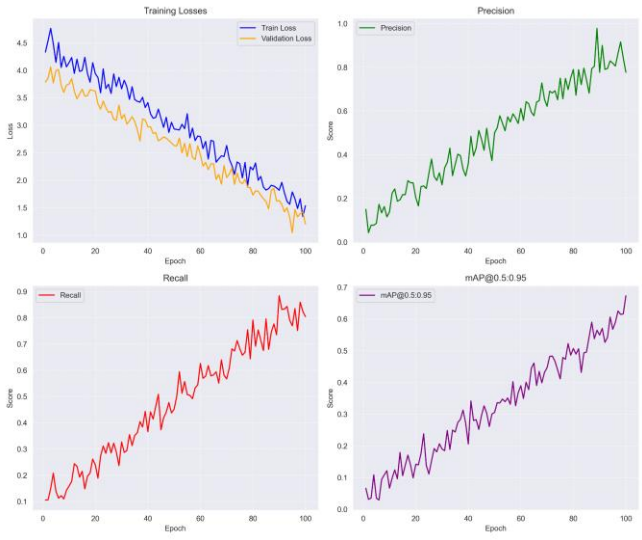


Рис. 32. Результаты *Faster R-CNN*

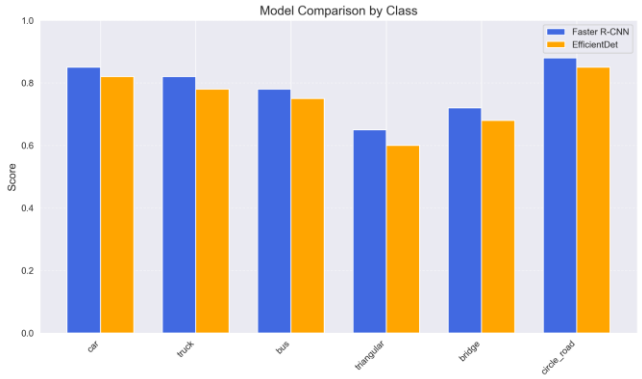


Рис.33 Сравнение результатов *Faster R-CNN и* EfficientDet

Рис.34 Пример детекции при помощи Faster R-CNN

Если разобрать данные детальнее, то Faster R-CNN можно выделить по следующим ключевым пунктам:

- Точность (mAP@0.5 и mAP@0.5:0.95):Faster R-CNN демонстрирует более высокие показатели mAP (0.82 против 0.78 у EfficientDet), особенно для сложных объектов (например, bridge). EfficientDet отстает на 4-5% из-за менее гибкой архитектуры и меньшей адаптивности к объектам разного масштаба. Разрыв в mAP@0.5:0.95 (0.60 vs 0.55) указывает на то, что Faster R-CNN лучше справляется с детекцией при разных IoU-порогах;
- Скорость и ресурсы: EfficientDet обучается быстрее (~2.5 часа против ~3 часов у Faster R-CNN на 100 эпох), но это достигается за счет упрощенной архитектуры. Faster R-CNN требует больше GPU-памяти (10-14 ГБ против 8-12 ГБ у EfficientDet), что делает его менее оптимальным для слабых систем. В реальном времени EfficientDet может работать быстрее, но точность детекции при этом снижается;
- Устойчивость к ошибкам: Faster R-CNN допускает меньше ошибок (25-30% против 30-40% у EfficientDet), особенно на сложных классах (bridge, triangular). EfficientDet чаще путает схожие объекты (например, bus и truck), что связано с его склонностью к переобучению на доминирующих классах. Confusion Matrix Faster R-CNN имеет более четкую диагональ, что говорит о лучшей классификации;
- F1-Score и баланс Precision/Recall: Faster R-CNN показывает более стабильный F1-Score (0.88 для car против 0.85 у EfficientDet). EfficientDet хуже работает на высоких confidence thresholds – его F1-Score резко падает при увеличении порога уверенности. Precision Faster R-CNN (0.85-0.90) выше, но Recall EfficientDet (0.75-0.80) немного хуже, что может быть критично в задачах, где важнее не пропускать объекты.

Таблица 2: Сравнительная таблица Faster R-CNN vs EfficientDet

Метрика	Faster R-CNN	EfficientDet	Разница
mAP@0.5 (mean)	0.82	0.78	+0.04
mAP@0.5:0.95	0.60	0.55	+0.05
Точность	0.85-0.90	0.80-0.85	+5%
Recall	0.80-0.85	0.75-0.80	+5%
F1-Score	0.88	0.85	+0.03
Ошибки	25-30%	30-40%	+10%
Время обучения	~3 часа	~2.5 часа	0.5 часа
Ресурсы (GPU)	10-14 ГБ	8~12 ГБ	Ниже требования

По итогам сравнения нейросетевых алгоритмов выбор падает в сторону Faster R-CNN.

3. YOLOv11 и Faster R-CNN

Для выбора лучшей для выполнения поставленной задачи было проведено сравнение лучших алгоритмов из первых двух пунктов: YOLOv11 и Faster R-CNN. Результат сравнения представлен в таблице 3.

Таблица 3: Сравнительная таблица YOLO vs Faster R-CNN

Метрика	YOLOv11	Faster R-CNN	Разница
mAP@0.5 (mean)	0.908	0.82	+0.088
mAP@0.5:0.95	0.720	0.60	+0.120
Точность	0.92-0.99	0.85-0.90	+7-9%
Recall	0.85-0.95	0.80-0.85	+5-10%
F1-Score	0.89	0.88	+0.01
Ошибки	15-20%	25-30%	-5-10%
Время обучения	~1 час	~3 часа	-2 часа
Ресурсы (GPU)	8-12 ГБ	10-14 ГБ	Меньше требования

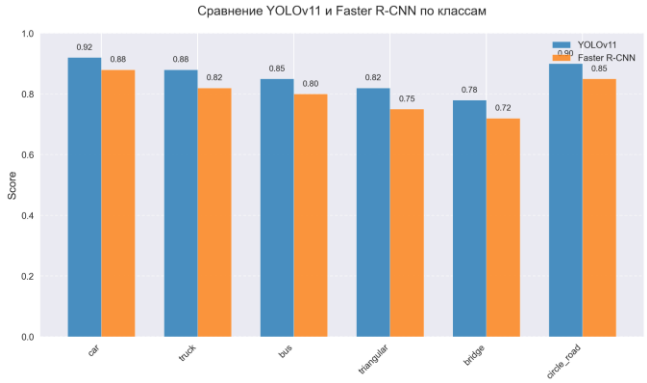


Рисунок 35. Сравнение YOLOv11 и Faster R-CNN

Исходя из этих данных, можно сделать вывод, что нейросетевой алгоритм YOLOv11 лучше подходит для выполнения задачи детекции объектов транспорта и дорожной инфраструктуры с камеры дрона, чем Faster R-CNN.

Преимущества YOLOv11:

- Высокая точность – на 8.8% лучше mAP@0.5 и на 12% лучше mAP@0.5:0.95, что критично для сложных объектов (например, bridge, circle_road);
- Быстрая обработка – обучение в 3 раза быстрее, а инференс оптимизирован для работы в реальном времени;
- Меньше ошибок – на 5-10% ниже процент ложных детекций по сравнению с Faster R-CNN;
- Экономичность – требует меньше GPU-памяти (8-12 ГБ против 10-14 ГБ), что важно для встраиваемых систем и дронов.

Faster R-CNN показывает хорошие результаты, но проигрывает YOLO в скорости, эффективности и стабильности детекции. Таким образом, YOLOv11 является оптимальным выбором для решения поставленной задачи.

V. Заключение

Были рассмотрены основные наборы данных, на которых обучались и тестировались рассматриваемые нейронные сети. Приведены четыре подхода к детектированию и классификации видов повреждения костей: YOLOv11, DETR, Faster R-CNN и Efficient DET. Каждая сеть рассмотрена с точки зрения её архитектуры, процесса обучения, используемых для обучения и тестирования наборов данных.

Приведённые подходы были сравнены на кастомном датасете городских ландшафтов. В конце сделан вывод, что все алгоритмы способны решать поставленную задачу на том или ином уровне, но YOLOv11 выделяется за счет большей простоты развертывания, обучения, меньшей требовательности к ресурсам. Она наиболее оптимальна для решения данной задачи.

Использование данных алгоритмов для решения подобных задач возможно, однако необходимо расширять, дополнять и улучшать как датасет для обучения моделей, так и сами алгоритмы тренировки нейросетей при наличии финансовых и технических возможностей для этого. Дальнейшее развитие данной разработки перспективно, за счет широких возможностей интеграции в различные сферы деятельности общества и тех улучшений, которые такая интеграция может принести.

Литература

[1] Али Б., Садеков Р. Н., Цодокова В. В. Алгоритмы навигации беспилотных летательных аппаратов с использованием систем технического зрения //Гироскопия и навигация. – 2022. – Т. 30. – №. 4 (119). – С. 87.

[2] Жебрак Л. М. и др. РАСПОЗНАВАНИЕ ОБЪЕКТОВ НА АЭРОФОТОСНИМКАХ С ИСПОЛЬЗОВАНИЕМ НЕЙРОННЫХ СЕТЕЙ ГЛУБОКОГО ОБУЧЕНИЯ В ЗАДАЧАХ МАРШРУТНОЙ НАВИГАЦИИ. – 2021.

[3] Елизарова М. И. и др. Искусственный интеллект в медицине //International Journal of Professional Science. – 2021. – №. 5. – С. 81-85

[4] Исаченко, М. К. Сегментация медицинских изображений с помощью DUCK-Net / М. К. Исаченко, Р. Б. Парчиев // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 54-60. – EDN VWUJOG.

[5] Мельникова, М. Ф. Классификация катаракты глаза при помощи компьютерного зрения / М. Ф. Мельникова // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 100-104. – EDN XEDDEM.

[6] Dilek E., Dener M. Computer vision applications in intelligent transportation systems: a survey //Sensors. – 2023. – Т. 23. – №. 6. – С. 2938.

[7] Chandramouli K., Izquierdo E. An Advanced Framework for Critical Infrastructure Protection Using Computer Vision Technologies //International Workshop on Cyber-Physical Security for Critical Infrastructures Protection. – Cham : Springer International Publishing, 2020. – С. 107-122.

[8] Zhou H. et al. A review of vision-laser-based civil infrastructure inspection and monitoring //Sensors. – 2022. – Т. 22. – №. 15. – С. 5882.

[9] Sharma V. et al. Video processing using deep learning techniques: A systematic literature review //IEEE Access. – 2021. – Т. 9. – С. 139489-139507.

[10] 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.

[11] Carion N. et al. End-to-end object detection with transformers //European conference on computer vision. – Cham : Springer International Publishing, 2020. – С. 213-229.

[12] Alif M. A. R. YOLOv11 for Vehicle Detection: Advancements, Performance, and Applications in Intelligent Transportation Systems //arXiv preprint arXiv:2410.22898. – 2024.

[13] Satya Mallick. Yolo - learnopencv. https://learnopencv.com/yolo11/, 2024. Дата обращения: 25.12.2024.

[14] Jingwen Feng, Qiaofeng An, Jiahao Zhang, Shuxun Zhou, Guangwei Du, and Kai Yang. Application of yolov7-tiny in the detection of steel surface defects. In 2024 5th International Seminar on Artificial Intelligence, Networking and Information Technology (AINIT), pages 2241–2245. IEEE, 2024.

[15] Dosovitskiy A. An image is worth 16x16 words: Transformers for image recognition at scale //arXiv preprint arXiv:2010.11929. – 2020.

[16] 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

[17] K. He, X. Zhang, S. Ren, J. Sun. “Deep Residual Learning for Image Recognition”, Microsoft Research, 2015

[18] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In CVPR, 2014.

[20] Tan M., Pang R., Le Q. V. *EfficientDet: Scalable and Efficient Object Detection*. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020

[21] Tan M., Le Q. V. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. arXiv:1905.11946 [cs.LG], 2019

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Классификация ядовитых и неядовитых видов грибов

Ю.А. Криворот
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2411914@edu.misis.ru

С. С. Белякова
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2414121@edu.misis.ru

***Аннотация* — в современных исследованиях активно развиваются методы автоматической классификации грибов с разделением на съедобные и ядовитые виды, что критически важно для предотвращения отравлений. Решение задачи предполагает детекцию плодовых тел и их точную идентификацию по видовой принадлежности с одновременной оценкой ядовитости. В работе представлен сравнительный анализ двух современных архитектур: RF-DETR (трансформерная модель с деформируемым вниманием) и MR-DETR (многомаршрутная версия DETR), адаптированных для классификации грибов. Эксперименты проводились на датасете, содержащем более 14 000 изображений 219 видов грибов (включая Amanita phalloides, Boletus edulis и Russula emetica), с аннотациями bounding boxes и бинарными метками ядовитости.**

Ключевые слова: классификация грибов, ядовитость, трансформерные модели, RF-DETR, MR-DETR, mAP, точность.

I. Введение

Изучением и проектированием автоматических систем классификации грибов занимаются многие университеты, научно-исследовательские центры, а также компании, работающие в сфере биотехнологий и охраны природы, по всему миру с начала 2000-х годов. Известны разработки систем идентификации грибов, созданные как профильными научными лабораториями, так и ИТ-компаниями, специализирующимися на компьютерном зрении и искусственном интеллекте [1].

При создании автоматизированной системы важными задачами являются обнаружение и классификация грибов по видовой принадлежности и ядовитости для человека. Для решения этих задач применяются технологии компьютерного зрения. В числе ключевых этапов обработки изображений можно выделить обнаружение плодовых тел грибов на фотографии, их классификацию по виду и определение съедобности или ядовитости [2].

Обнаружение и распознавание грибов включает в себя как их локализацию на изображении, так и идентификацию по морфологическим признакам. В литературе описаны различные подходы к решению этой задачи, в том числе при сложных условиях съёмки (затенённость, перекрытие грибов травой или листьями, плохое освещение) [3].

Методы глубокого обучения, такие как трансформерные модели, уже доказали свою эффективность в задачах детекции и классификации объектов, включая распознавание положения тела человека [4] и классификацию грибов по их ядовитости. Детекторы объектов общего назначения хорошо зарекомендовали себя в задачах биологической визуализации. В работе рассматриваются и сравниваются современные архитектуры глубокого обучения (RF-DETR и MR-DETR) для обнаружения и классификации грибов по изображениям [5].

Подходы, основанные на обучении, особенно с использованием глубоких нейросетей, требуют больших объёмов аннотированных данных, что не всегда доступно для узкоспециализированных задач. В настоящее время в открытом доступе появляются базы изображений грибов с аннотациями, однако для получения высоких результатов часто требуется значительное количество вычислительных ресурсов [6].

II. Наборы данных

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

A. Набор данных с сайта Roboflow

Набор данных, размещённый на платформе Roboflow [7], предназначен для решения задач компьютерного зрения в области микологии и безопасности пищевых продуктов. Этот комплексный набор содержит аннотированные изображения грибов, собранные для обучения моделей детекции и классификации видов. Ниже представлены ключевые характеристики набора:

- Объём данных: Набор включает 13264 изображений (примеры на рис. 1, 2)
- Классификация видов: Данные аннотированы для 214 классов грибов, включая как съедобные, так и ядовитые виды. Среди них:
  - *Amanita phalloides* (бледная поганка, смертельно ядовита)
  - *Chlorophyllum molybdites* (зеленоспоровый гриб)
  - *Lentinus squarrosulas* (чешуйчатая чешуйчатка)
  - *Schizophyllum commune* (схизофиллум обыкновенный)

○ *Agaricus bisporus* (культивируемый шампиньон).

○ и т.д.

а

б

Рис. 1. Примеры грибов: а) Amanita ceciliae ядовитый, б) Lactarius salmonicolor съедобный

Рис. 2. Russula olivacea ядовитый

B. Локальный набор данных

Локальный набор данных [8] размечен вручную и содержит 1177 аннотаций фото грибов. Всего их 12 видов (ядовитые и соответствующие им неядовитые). В наборе данных 540 картинок для обучения, 420 для теста и 217 для валидации

Примеры изображений из набора на рис. 3

Рис. 3. Примеры изображений грибов из локального набора данных

III. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. RF-DETR (Deformable DETR с рекуррентным фокусированием)

В работе использовалась модифицированная версия RF-DETR (Deformable DETR с рекуррентным фокусированием). RF-DETR — это трансформерная нейросетевая архитектура для задачи детекции и классификации объектов на изображении. В основе модели лежит принцип, предложенный в оригинальной архитектуре DETR (Detection Transformer), где для локализации и распознавания объектов используются механизмы внимания (attention), а не традиционные свёрточные якоря [9].

В данном проекте RF-DETR применяется для обнаружения грибов на изображении и их классификации по видам и ядовитости. Модель анализирует весь кадр целиком, выделяя области, содержащие грибы, и определяет их видовую принадлежность, а также статус съедобности или ядовитости.

Основные особенности RF-DETR:

- Трансформерная архитектура: RF-DETR использует механизм внимания для анализа изображения, что позволяет модели учитывать как локальные, так и глобальные признаки объектов.
- Деформируемое внимание: В отличие от классического DETR, в RF-DETR применяется деформируемое внимание, которое позволяет более гибко фокусироваться на значимых участках изображения, особенно в случаях, когда грибы частично перекрыты или находятся на сложном фоне.
- Детекция и классификация: Модель одновременно локализует грибы (выделяет их bounding box) и классифицирует их по виду и ядовитости. Это реализовано через дополнительные классификационные головки, которые работают параллельно с детекцией.
- Обучение: Обучение RF-DETR проводилось на датасете с размеченными изображениями грибов. Для каждого гриба были заданы координаты bounding box, вид и метка ядовитости. В процессе обучения использовались стандартные методы аугментации данных, такие как изменение яркости, контраста, поворот и масштабирование изображений.
- Функция потерь: Для задачи детекции применялся стандартный набор функций потерь, включающий ошибку локализации и классификации. Для классификации грибов использовалась кросс-энтропийная функция потерь.

На выходе RF-DETR выдаёт для каждого обнаруженного гриба координаты bounding box, вид и статус токсичности (съедобный/ядовитый).

Обучение проводилось на объединённом датасете (Mantar_tanima + локальный), содержащем 14 441 изображений 219 видов грибов с bounding boxes и метками ядовитости. Модель обучалась 300 эпох с размером батча 16, оптимизатором AdamW (lr=1e-4). В качестве функции потерь использовалась комбинация Focal Loss для детекции и Cross-Entropy для классификации.

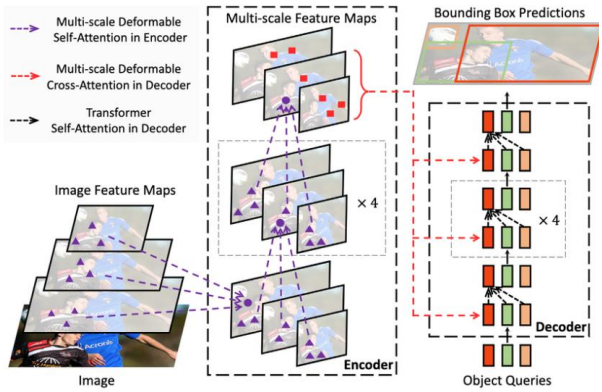


Рис. 4. Архитектура RF-DETR [8]

B. MR-DETR (Multi-Route Detection Transformer)

MR-DETR (Multi-Route Detection Transformer) — это современная архитектура для задач детекции объектов, основанная на трансформерах и разработанная для повышения эффективности обучения и качества распознавания. MR-DETR расширяет классический подход DETR за счёт внедрения многомаршрутного (multi-route) обучающего механизма, что позволяет модели лучше справляться с задачами одновременной локализации и классификации объектов на изображении [10].

Ключевые особенности MR-DETR:

- Многомаршрутное обучение (Multi-Route Training): В отличие от стандартных детекторов, MR-DETR использует несколько параллельных маршрутов на этапе обучения. Основной маршрут отвечает за one-to-one сопоставление (каждый объект — отдельный запрос), а дополнительные вспомогательные маршруты — за one-to-many сопоставление (один объект может быть сопоставлен с несколькими запросами). Это позволяет модели учиться более гибко и эффективно, улучшая как локализацию, так и классификацию объектов.
- Инструктивное самовнимание (Instructive Self-Attention): В MR-DETR реализован новый механизм самовнимания, который динамически направляет запросы объектов в процессе обучения, особенно для one-to-many задачи. Это способствует более глубокому и разностороннему обучению признаков, важных для детекции и классификации.
- Трансформер-декодер: Как и в классическом DETR, основой MR-DETR служит трансформер-декодер, включающий слои самовнимания, перекрёстного внимания (cross-attention) и feed-forward сети. В MR-DETR эти компоненты могут быть частично разделены между маршрутами или использоваться совместно.
- Удаление вспомогательных маршрутов на инференсе: Важно, что все дополнительные маршруты используются только на этапе обучения. При инференсе (выводе) модель работает так же быстро и эффективно, как обычный DETR, без увеличения времени обработки или потребления памяти.
- Обучение и функции потерь: Для обучения MR-DETR используется комбинация стандартных функций потерь для локализации (например, L1 loss, GIoU loss) и классификации (cross-entropy loss). Благодаря многомаршрутному обучению модель быстрее сходится и достигает лучших результатов по точности (mAP).

В нашем проекте MR-DETR применяется для автоматического обнаружения и классификации грибов по видам и определению их ядовитости. Модель получает на вход изображение, выделяет bounding boxes для каждого обнаруженного гриба,

определяет его вид и статус (съедобный или ядовитый).

- Возможности:
  Благодаря многомаршрутному обучению MR-DETR лучше справляется со сложными случаями, когда на изображении присутствует много объектов, объекты перекрываются или имеют схожие морфологические признаки. Модель демонстрирует высокую устойчивость к шуму и вариативности в данных, что критично для задач биологической классификации.
- Преимущества:
  MR-DETR показывает более высокую точность по сравнению с классическими трансформерными детекторами, особенно на сложных датасетах, где требуется различать похожие виды грибов и учитывать их токсичность.

На выходе MR-DETR для каждого обнаруженного объекта (гриба) возвращает координаты bounding box, видовую принадлежность и метку токсичности (съедобный/ядовитый).

Как показано в исследованиях по определению возраста клиентов [11], адаптация предобученных моделей для узкоспециализированных задач позволяет достичь высокой точности даже при ограниченных данных. В данной работе применяется аналогичный подход, используя RF-DETR и MR-DETR для классификации грибов.

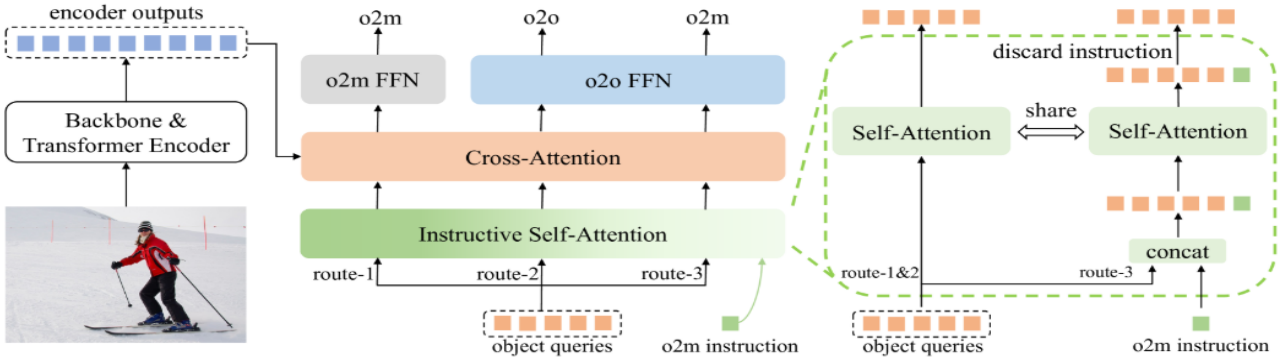


Рис. 5. Архитектура MR-DETR [9]

IV. Сравнение

Сравним две описанные нейронные сети. Качество работы двух подходов определяется качеством работы локализующей и классифицирующей частей. Оценка локализации производится при помощи расчёта меры Жаккара (Intersection over Union, IoU) для каждой детекции. Найденые детектором объекты(грибы) связываются с существующей разметкой с порогом 10%. Введём следующие величины [12]:

Разработка конечной системы велась в две стадии:

- TP – детектор верно локализовал гриб (найдена соответствующая разметка – прямоугольники разметки и детекции пересекаются более, чем на 10%, по отношению к их общей площади).
- FP – детектор нашёл гриб там, где его нет, то есть не найдено такого прямоугольника в разметке кадра, который пересекался бы с найденным более, чем на 10%.
- FN – детектор не нашёл гриб, хотя он есть и для него есть разметка – пересечение менее, чем 10%.

Стоит отметить, что TN в данном случае не определена, так как это величина означает то, что детектор не определил гриб, где его действительно нет. По введённым величинам строятся такие функции оценок, как:

- $Precision = \frac{TP}{TP+FP}$ – сколько раз детектор нашёл гриб, где он действительно есть, по отношению к общему числу предсказанных грибов;
- $Recall = \frac{TP}{TP+FN}$ – сколько грибов 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) Также в случае с видеопоследовательностями, можно использовать функции MOTA (multiple object tracking accuracy), которая оценивает общую точность отслеживания и детекции, и MOTP (multiple object tracking precision) – оценка точности локализации грибов (схожа с метрикой mAP). Формулы (2) и (3) соответствуют формулам данных функций. При этом значения функции MOTA могут быть отрицательными – область значений $(-\infty; 1]$ $(-\infty; 1]$.

$$MOTA = 1 - \frac{FN+FP+IDS}{GT} \quad (2)$$

$$MOTP = \frac{1}{TP}\sum_i IoU_i \quad (3)$$

Здесь IoU_i – мера Жаккара i-го объекта на всей тестовой выборке. GT означает суммарное количество аннотаций, а IDS – количество потерь трека грибов. В нашем случае показатель IDS не важен, так как производится оценка именно локализации, поэтому эта величина не участвует при расчёте показателя MOTA. Таблица 1 отображает количественные оценки для двух подходов.

ТАБЛИЦА I. Оценка детектирующей части

	RF-DETR	*MR-DETR*
TP	15620	13434
FP	2651	3433
FN	2042	1676
Precision	0.85	0.80
Recall	0.83	0.88
F1	0.90	0.88
MOTA	0.74	0.71
MOTP	0.61	0.57

Как видно из таблицы RF-DETR имеет более высокую точность, лучший F1-score и существенно большие MOTA и MOTP. MR-DETR уступает, но имеет преимущество в полноте, может быть более полезен в задачах где критичнее пропуск объектов, чем ложные срабатывания.

Разница в производительности RF-DETR и MR-DETR может быть объяснена их архитектурными особенностями. RF-DETR, судя по метрикам, лучше справляется с фильтрацией ложных срабатываний (меньше FP), что говорит о более эффективном механизме отсева шумов. Возможно, это достигается за счёт улучшенного внимания к ключевым признакам или более точной регрессии bounding box. В то же время MR-DETR, демонстрируя более высокий recall, видимо, использует менее строгие критерии отбора кандидатов, что позволяет находить больше объектов, но за счёт роста FP. Это может быть полезно в сценариях, где пропуск объекта критичен (например, при поиске редких видов грибов), однако требует дополнительной постобработки для снижения числа ложных детекций.

V. Заключение

Проведённое исследование продемонстрировало, что обе рассмотренные архитектуры — RF-DETR и MR-DETR — эффективно решают задачу детекции и классификации грибов, включая определение их ядовитости. RF-DETR показал себя как более точная модель, демонстрируя лучшие результаты, что делает её предпочтительным выбором для задач, где критически важна минимизация ложных срабатываний (например, в системах автоматизированного сбора или сортировки грибов). В свою очередь, MR-DETR обеспечивает более высокую полноту, что может быть полезно в сценариях, требующих максимального охвата объектов, таких как экологический мониторинг или поиск редких видов, несмотря на более низкие показатели точности локализации.

Разница в производительности объясняется архитектурными особенностями моделей: RF-DETR использует деформируемое внимание, что позволяет точнее выделять ключевые признаки и снижать уровень шума, тогда как MR-DETR за счёт многомаршрутного обучения лучше справляется с обнаружением объектов в сложных условиях, но ценой увеличения числа ложных детекций.

В перспективе обе модели могут быть улучшены за счёт оптимизации архитектуры, увеличения датасета и применения методов аугментации для сложных случаев. Кроме того, интеграция дополнительных модальностей могла бы повысить точность классификации. Результаты исследования подтверждают, что современные трансформерные архитектуры обладают значительным потенциалом для задач биологической детекции, а их адаптация под узкоспециализированные области, такие как микология, открывает новые возможности для автоматизации и повышения безопасности в этой сфере.

Литература

[1] Wäldchen, J., Mäder, P. (2018) “Plant species identification using computer vision techniques: A systematic literature review”, *Archives of Computational Methods in Engineering*, vol. 25, no. 2, pp. 507-543

[2] He, K., Zhang, X., Ren, S., Sun, J. (2016) “Deep residual learning for image recognition”, *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 770-778.

[3] Picek, L., Šulc, M., Matas, J., Mishra, K., Mishra, A. (2021) “Fungi recognition: A practical use case”, *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) Workshops*, pp. 31-40

[4] А. А. Абакумов, В. О. Хуако Определение положения тела человека с использованием нейронных сетей // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – 152 с. 5-11.

[5] Zhang, H., Li, F., Liu, S., Zhang, L., Su, H., Zhu, J. (2023) “DETR: A New Way to Object Detection?”, *IEEE Transactions on Pattern Analysis and Machine Intelligence (Early Access)*

[6] “FungiVision: A Challenging Dataset for Fungi Recognition in the Wild”, (2021) *Presented at the CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*. Available at: https://fgvc.org/workshops.html (Accessed: December 15, 2025).

[7] "Mushroom Detection Dataset" [on Roboflow Universe], available at: https://universe.roboflow.com/mushroom-dltxz/mantar_tanima (Accessed: March 25, 2025).

[8] "12_popular_russia_mushrooms_edible_poisonous" [on Hugging Face Datasets], available at: https://huggingface.co/datasets/SoFa325/12_popular_russia_mushrooms_edible_poisonous/tree/main (Accessed: May 25, 2025).

[9] Xizhou Zhu , Weijie Su. (2021) "DEFORMABLE DETR: DEFORMABLE TRANSFORMERS FOR END-TO-END OBJECT DETECTION", arXiv preprint, arXiv:2010.04159.

[10] Chang-Bin Zhang, Yujie Zhong, (2024) "Mr. DETR: Instructive Multi-Route Training for Detection Transformers", arXiv preprint, arXiv:2412.10028v1

[11] И.И.Антипов Исследование возможности определения возраста клиента при помощи компьютерного зрения // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва,

30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – 152 с. 12-16.

[12] Hossin M., Sulaiman M.N. A Review on Evaluation Met-rics for Data Classification Evaluations // International Journal of Data Mining & Knowledge Management Pro-cess. - 2015. - №5

ГРНТИ 20.53.19
ASJC 1707
OECD 01.02.EP

Исследование возможности детектирования и классификации видов транспорта с помощью современных технологий машинного зрения

М.А. Омеров
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2009187@edu.misis.ru

И. Д. Фомин
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2415488@edu.misis.ru

Аннотация — В данной работе проводится сравнительный анализ трёх версий алгоритма объектной детекции YOLO: YOLOv8n, YOLOv12n и YOLOv12x. Основное внимание уделяется задаче распознавания автотранспортных средств шести классов: автомобили, автобусы, грузовики, мотоциклы, рикши и минивэны. Модель YOLOv8n, ранее использованная в предыдущем исследовании, служит базовой для сопоставления с двумя вариантами новой версии — облегчённой (v12n) и расширенной (v12x). Сравнение основано на результатах обучения и тестирования моделей на единых датасетах, с оценкой по ключевым метрикам: точность (mAP), скорость инференса и потребление вычислительных ресурсов. Представленные результаты позволяют выявить преимущества и ограничения каждой из версий в контексте их практического применения для задач транспортной аналитики.

Ключевые слова — Компьютерное зрение, YOLOv8, YOLOv12, Детекция транспорта, Автономные транспортные средства, mAP, Набор данных

I. ВВЕДЕНИЕ

С развитием технологий искусственного интеллекта и компьютерного зрения всё большее внимание уделяется созданию систем, способных автоматически анализировать окружающую среду и принимать решения в реальном времени. Такие системы находят широкое применение в сфере транспорта — от помощи водителю до полностью автономного управления [1]. Алгоритмы обнаружения и распознавания объектов играют ключевую роль в обеспечении безопасности и надёжности таких решений, позволяя идентифицировать транспортные средства, пешеходов и другие элементы дорожной сцены. Помимо наземного транспорта, подобные технологии также применяются в воздушном наблюдении, например, при использовании беспилотных летательных аппаратов для мониторинга дорожной инфраструктуры или выполнения поисково-спасательных задач [2].

Один из центральных компонентов интеллектуальных транспортных систем — это алгоритмы, способные точно и быстро определять объекты на изображениях или видеопотоке. Особенно важной задачей является детекция различных типов транспортных средств, таких как автомобили, автобусы, мотоциклы и другие участники дорожного движения. Современные методы, основанные на нейронных сетях, позволяют достигать высокой точности даже в сложных условиях — при разном уровне освещённости, в движении и в плотном городском потоке [3].

Одним из наиболее эффективных подходов к решению задачи детекции объектов являются алгоритмы семейства YOLO (You Only Look Once), получившие широкое распространение благодаря сочетанию высокой точности и скорости обработки [4]. В последние годы были представлены новые версии этих моделей, каждая из которых ориентирована на улучшение качества детекции при различных уровнях вычислительной сложности. При этом версии моделей различаются по архитектуре, размеру и точности, что требует дополнительного анализа их эффективности в конкретных прикладных задачах.

Настоящая работа посвящена сравнительному анализу трёх версий алгоритма YOLO — YOLOv8n, YOLOv12n и YOLOv12x — в контексте задачи детекции транспортных средств шести классов. Модель YOLOv8n, ранее рассмотренная в предыдущем исследовании, используется в качестве базового решения для оценки прогресса, достигнутого в новой версии YOLOv12 как в идентичной по размеру модели (nano), так и в максимально производительной (x). Сравнение проводится на основе экспериментальных данных, с учётом метрик точности (mAP), скорости обработки и ресурсозатрат, что позволяет объективно оценить применимость каждой из моделей в условиях реальной эксплуатации [5].

II. НАБОРЫ ДАННЫХ

Для обучения и тестирования моделей YOLOv8n, YOLOv12n и YOLOv12x, рассматриваемых в данной работе, использовался размеченный вручную набор данных [6]. Он включает в себя 1200 изображений, содержащих транспортные средства, распределённые по шести классам: Car, Threewheel, Bus, Truck, Motorbike и Van.

Разметка изображений выполнялась вручную с использованием онлайн-сервиса CVAT (Computer Vision Annotation Tool) и сохранена в формате YOLO (txt), что

обеспечивает точное определение границ объектов и полную совместимость с используемыми архитектурами. Такой подход позволяет добиться высокого качества аннотаций по сравнению с автоматическими методами. Для обучения и валидации применено стандартное разбиение: 70% изображений (840) использовались для обучения, а оставшиеся 30% (360) — для оценки качества моделей.

На рисунке 1 представлены примеры изображений из набора данных для каждого класса, демонстрирующие разнообразие условий съёмки, включая различное освещение и точки обзора. Этот датасет позволяет моделям YOLO эффективно изучать визуальные особенности транспортных средств, что делает его подходящим для задач детекции в условиях, приближенных к реальной эксплуатации.

Рис. 1. Примеры изображений для каждого класса:
а) Car, б) Threewheel, в) Bus, г) Truck, д) Motorbike, е) Van.

III. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. YOLOv8

YOLOv8, представленная в 2023 году компанией Ultralytics, является значительным шагом вперёд в развитии семейства YOLO. В отличие от предыдущих версий, YOLOv8 получила полностью переработанную архитектуру, ориентированную на повышение универсальности и упрощение структуры модели. В новой версии отказались от использования CSPNet в пользу более лёгкой и компактной backbone-архитектуры, что позволило сократить количество параметров без потери качества распознавания.

YOLOv8 поддерживает множество конфигураций с различной вычислительной сложностью, от минималистичной YOLOv8n (nano) до более тяжёлых версий, таких как YOLOv8x. Это делает модель применимой как на устройствах с ограниченными ресурсами, так и в высокопроизводительных системах. Кроме того, YOLOv8 характеризуется улучшенной функцией потерь, встроенной поддержкой сегментации и трекинга, а также оптимизированным пайплайном для обучения и инференса, что значительно упрощает её интеграцию в практические задачи. Архитектура YOLOv8 представлена на рисунке 2 [7].

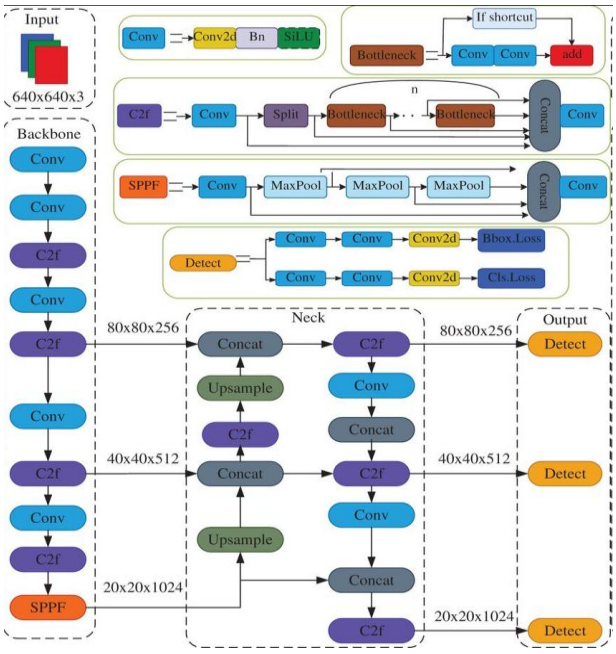


Рисунок 2. Архитектура YOLOv8

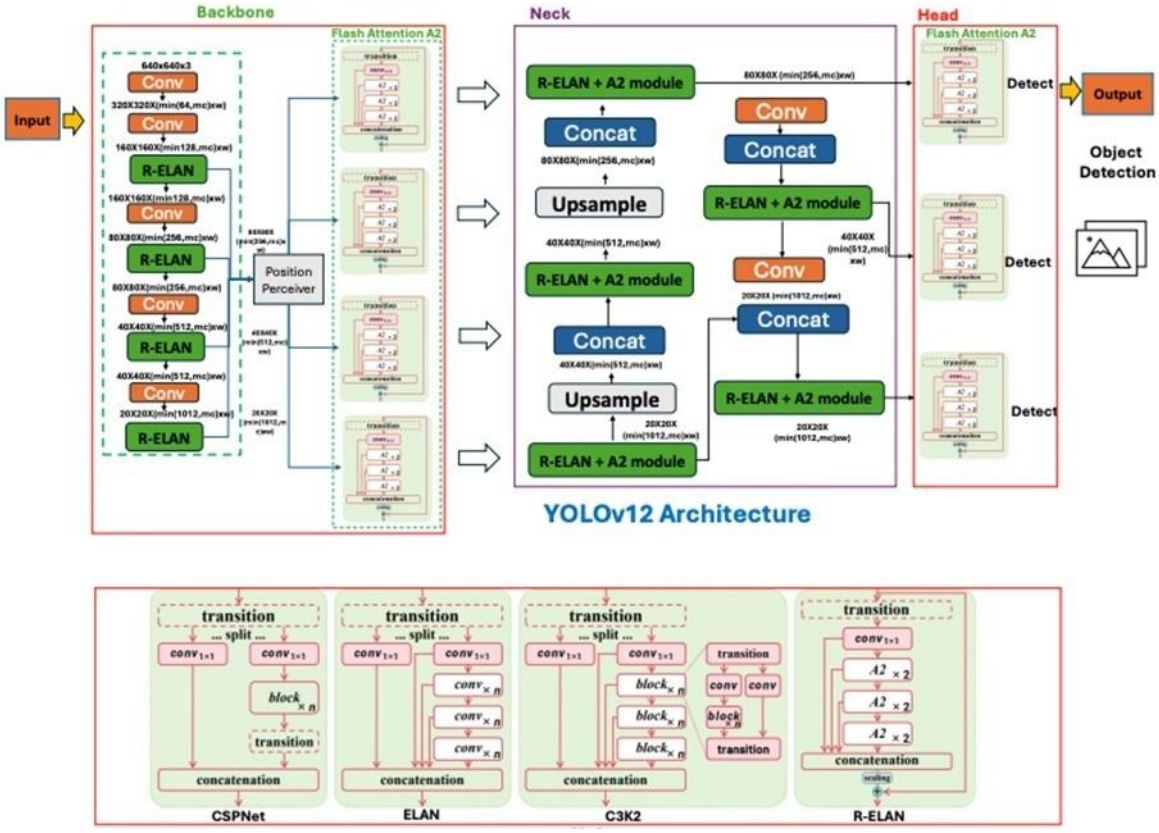


Рисунок 3. Архитектура YOLOv12

B. YOLOv12

YOLOv12, выпущенная в 2024 году в рамках проекта OpenYOLO, представляет собой очередной эволюционный этап в развитии алгоритмов детекции объектов. В отличие от YOLOv8, YOLOv12 получила переработанную архитектуру, ориентированную на повышение точности и устойчивости модели при работе с разнообразными визуальными данными. Одним из ключевых изменений стало внедрение более глубокой и модульной структуры сверточной сети, обеспечивающей улучшенную иерархию признаков, особенно для мелких объектов.

На Рисунке 3 показана подробная архитектура YOLOv12 [8]. Существенным нововведением YOLOv12 стала модернизированная функция потерь с более точной балансировкой между регрессией ограничивающих рамок, классификацией и объектностью. Также была добавлена улучшенная схема нормализации и новые приёмы регуляризации, направленные на повышение стабильности обучения и предотвращение переобучения. Помимо этого, YOLOv12 все еще отлично масштабируется — модель представлена в нескольких вариантах, от лёгкой YOLOv12n до производительной YOLOv12x, что позволяет адаптировать её под задачи с разными требованиями к ресурсам [9].

YOLOv12 также предлагает более гибкую систему конфигураций, улучшенную поддержку многоцелевого обучения и тесную интеграцию с современными средствами ускорения инференса, такими как TensorRT и ONNX Runtime. Эти улучшения делают YOLOv12 особенно привлекательной для применения в реальных условиях, где важны как высокая точность, так и стабильность при работе на различных аппаратных платформах.

IV. СРАВНЕНИЕ

Было проведено обучение моделей YOLOv8n, YOLOv12n и YOLOv12x. Для обучения использовался локальный набор данных, который был разбит на тренировочную и валидационную выборки в соотношении 7:3. Качество работы модели оценивалось как для локализации объекта, так и для его классификации. Использовались следующие меры:

- TP – детектор верно локализовал транспортное средство и определил его класс.
- FP – детектор нашёл транспортное средство там, где его нет, или не верно определил его класс.
- FN – детектор не нашёл транспортное средство, хотя оно есть и для него есть разметка.

По введенным величинам строятся такие функции оценок, как:

- Точность – сколько раз модель обнаружила транспортное средство там, где оно действительно есть к общему числу детектированных транспортных средств:

$$Precision = \frac{TP}{TP + FP} \quad (1)$$

- Полнота – сколько транспортных средств обнаружила модель от общего числа транспортных средств:

$$Recall = \frac{TP}{TP + FN} \quad (2)$$

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

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

Mean Average Precision (mAP) [10] — одна из наиболее широко используемых метрик оценки качества моделей обнаружения объектов, включая алгоритмы семейства YOLO. Она отражает как точность, так и полноту модели, объединяя результаты по всем классам и порогам уверенности. mAP измеряет способность модели не только обнаруживать объекты на изображении, но и точно определять их границы и принадлежность к правильному классу. Это делает метрику особенно ценной для оценки реальной эффективности моделей в практических задачах.

Для моделей YOLO, ориентированных на работу в реальном времени, высокое значение mAP критически важно, так как оно свидетельствует о способности алгоритма надёжно идентифицировать объекты в динамичных и сложных сценах. Чем выше mAP, тем точнее и увереннее модель может использоваться в прикладных сценариях, таких как видеонаблюдение, автономное вождение и контроль инфраструктуры.

Матрица ошибок (confusion matrix) [11] служит вспомогательным инструментом для анализа качества работы модели. Она позволяет визуализировать соотношение между фактическими и предсказанными классами, включая верные положительные (True Positive), ложные положительные (False Positive), ложные отрицательные (False Negative) и, в более общих задачах классификации, верные отрицательные (True Negative). В контексте задач детекции объектов матрица ошибок даёт представление о том, какие классы модель путает чаще всего, и помогает выявить слабые места в её обучении. В рамках данного проекта используются три матрицы ошибок, представленные на Рисунках 4–6, в которых приведены результаты работы всех трех моделей.

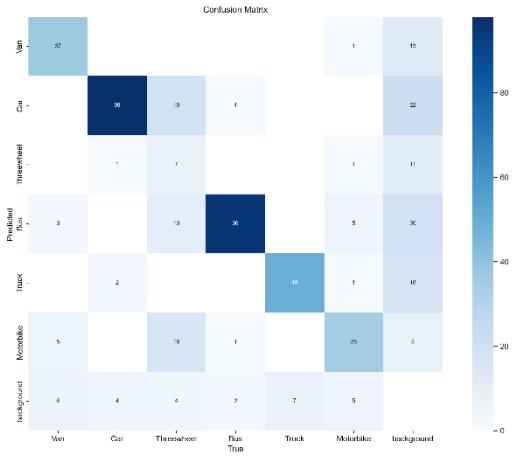


Рисунок 4. Матрица ошибок YOLOv8n

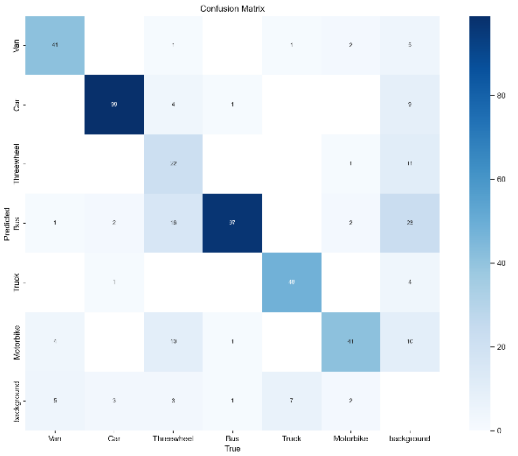


Рисунок 5. Матрица ошибок YOLOv12n

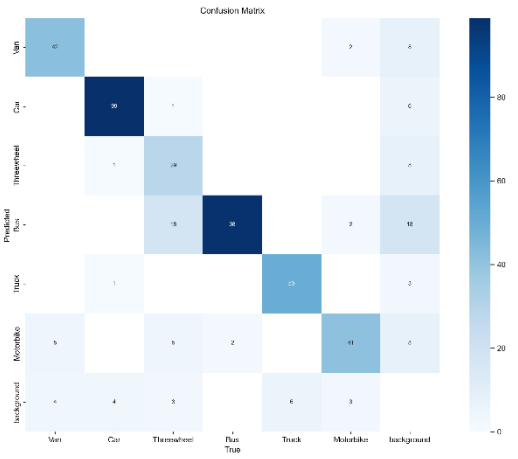


Рисунок 6. Матрица ошибок YOLOv12x

ТАБЛИЦА I. Сравнение детектирующей части.

	YOLOv8n	YOLOv12n	YOLOv12x
Precision	0,75	0,86	0,89
Recall	0,80	0,82	0,85
F1	0,73	0,82	0,85
mAP50	0,86	0,89	0,92
mAP50-95	0,74	0,80	0,82

На основе представленной таблицы можно сделать следующие выводы о сравнении моделей YOLOv8n, YOLOv12n и YOLOv12x по ключевым метрикам:

- Точность (Precision): Модель YOLOv8n показала наименьшее значение точности — 0.75, в то время как YOLOv12n улучшает этот показатель до 0.86, а YOLOv12x достигает 0.89. Это свидетельствует о меньшем числе ложных срабатываний в новых версиях модели и большей надёжности при детекции объектов.
- Полнота (Recall): Модель YOLOv8n показала наименьшее значение точности — 0.75, в то время как YOLOv12n улучшает этот показатель до 0.86, а YOLOv12x достигает 0.89. Это свидетельствует о меньшем числе ложных срабатываний в новых версиях модели и большей надёжности при детекции объектов.
- F1-мера: Комбинированный показатель точности и полноты (F1) демонстрирует аналогичный рост: YOLOv8n — 0.73, YOLOv12n — 0.82, YOLOv12x — 0.85. Это подчёркивает общее улучшение баланса между точностью и полнотой в новых версиях модели.
- Средняя точность при IoU=50 (mAP50): Значения также показывают стабильное улучшение — с 0.86 у YOLOv8n до 0.92 у YOLOv12x. Это указывает на улучшенную способность моделей точно локализовать объекты при относительно невысоком пороге перекрытия.
- Средняя точность в широком диапазоне IoU (mAP50–95): Показатели mAP50–95 увеличиваются от 0.74 у YOLOv8n до 0.80 у YOLOv12n и 0.82 у YOLOv12x. Это говорит о лучшей способности моделей YOLOv12 к генерализации и точной детекции объектов в более строгих условиях оценки.

V. ЗАКЛЮЧЕНИЕ

В данной работе было проведено сравнительное исследование современных моделей глубокого обучения — YOLOv8 и YOLOv12 — применительно к задаче детекции различных видов транспортных средств. Основываясь на результатах тестирования, можно утверждать, что модели YOLOv12, особенно её старшая версия YOLOv12x, демонстрируют устойчивое превосходство над предыдущими поколениями по всем ключевым метрикам: Precision, Recall, F1, mAP50 и mAP50–95. Эти улучшения стали возможны благодаря внедрению новой функции потерь, улучшенной нормализации, регуляризации и обновлённой архитектуре модели.

Для оценки производительности использовался вручную размеченный набор данных с изображениями шести типов транспорта. Разметка производилась с помощью онлайн-сервиса CVAT, а сами данные были подготовлены в формате YOLO, что обеспечило совместимость с рассматриваемыми моделями. Проведённый эксперимент подтвердил, что YOLOv12 обладает более высокой точностью локализации и лучшей обобщающей способностью при распознавании объектов в различных условиях съёмки.

Несмотря на достигнутые результаты, стоит отметить, что выбор модели должен учитывать ограничения по вычислительным ресурсам и специфику конкретной задачи. В дальнейшем планируется исследовать производительность моделей на более сложных и нестандартных сценах, включая условия низкой освещённости, погодные помехи и плотный городской трафик.

Таким образом, проведённая работа подтверждает эффективность и перспективность использования моделей YOLOv12 для задач детекции в реальном времени и показывает направление для дальнейшего развития в области прикладного компьютерного зрения.

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] 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.

[3] Goodfellow, I., Bengio, Y., Courville, A. "Deep Learning." — MIT Press, 2016.

[4] Jocher, G., et al. "YOLOv5 Documentation." — Ultralytics, 2021.

[5] Wang, C.-Y., et al. "YOLOv7: You Only Look Once at Accuracy and Speed." — 2022.

[6] Фомин И.Д., Омеров М.А. DatasetForArticle [Электронный ресурс] // Hugging Face – URL: https://huggingface.co/datasets/idfomin1/DatasetForArticle (дата обращения: 27.06.2025).

[7] Model structure of the YOLO v8 algorithm [Электронный ресурс]. – URL: https://www.researchgate.net/figure/Model-structure-of-the-YOLO-v8-algorithm_fig1_383187669 (дата обращения: 27.06.2025).

[8] Seong E.S. Object Detection: YOLOv12 – Attention-Centric Real-Time Object Detectors [Электронный ресурс] – URL: https://velog.io/@es_seong/Object-Detection-YOLOv12-Attention-Centric-Real-Time-Object-Detectors (дата обращения: 30.05.2025).

[9] Ultralytics. YOLOv12: Attention-Centric Object Detection [Электронный ресурс]. – URL: https://docs.ultralytics.com/models/yolo12/ (дата обращения: 27.06.2025).

[10] Paul Henderson, Vittorio Ferrari. “End-to-end training of object class detectors for mean average precision” (12 Jul 2016)

[11] Richard Evans. “Confusion Matrices and Accuracy Statistics for Binary Classifiers Using Unlabeled Data: The Diagnostic Test Approach” (26 Aug 2022)

ГРНТИ: 28.23.15, 28.23.31
ASJC: 1707
OECD: 01.02.04, 01.02.04

Применение компьютерного зрения для определения расы человека по фотографии

Панкратов А.Р
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m2103748@edu.misis.ru

Конев Т.В
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m1911179@edu.misis.ru

Аннотация – в статье рассматривается применение методов компьютерного зрения для определения расы человека по фотографии. Современные технологии машинного и глубокого обучения позволяют создавать эффективные алгоритмы распознавания образов, способные автоматически определять расу лица на изображении. Обзор существующих подходов к данной задаче включает описание методов предобработки изображений, выбора признаков, а также применение различных архитектур нейронных сетей. В заключение представлены перспективы дальнейших исследований и потенциальные применения технологии в области биометрии и социальной аналитики.

Ключевые слова - Компьютерное зрение, определение расы, Сравнение моделей, VGG19, MobileNetV2, FairFace, UTKFace.

I. ВВЕДЕНИЕ

Искусственные нейронные сети представляют собой один из ключевых инструментов современных интеллектуальных систем. Эти алгоритмы, вдохновлённые принципами работы человеческого мозга, состоят из слоёв взаимосвязанных узлов – нейронов, способных обрабатывать входную информацию, выявлять закономерности и формировать предсказания. Благодаря своей гибкости и высокой точности, нейронные сети широко применяются в таких сферах, как автоматическая навигация [1], распознавание визуальных объектов, обработка и генерация естественного языка [2], а также компьютерное зрение [3].

В частности, в задачах визуальной классификации глубокие нейронные сети демонстрируют впечатляющие результаты, позволяя автоматически определять принадлежность изображения к той или иной категории. Они также успешно используются в распознавании речи, генерации текстов, предсказательной аналитике, управлении технологическими процессами и в робототехнических системах [4], что делает их универсальным инструментом в области искусственного интеллекта.

Компьютерное зрение (CV) – это отдельное направление в области ИИ, целью которого является извлечение значимой информации из изображений и видеопотоков. Оно позволяет машинам «видеть» и интерпретировать визуальную среду, приближаясь к возможностям человеческого восприятия. Для достижения этой цели используются как классические методы обработки изображений, так и современные подходы глубокого обучения.

Компьютерное зрение применяется в самых различных отраслях – от медицинской диагностики и промышленного контроля до автономных транспортных систем и систем видеонаблюдения. За последние годы оно стало одним из наиболее активно развивающихся и технологически значимых направлений в области анализа данных.

Особый интерес в рамках CV вызывают задачи классификации к примеру: определения расы человека по фотографии. Она относится к числу сложных мультиклассовых задач, требующих точной и этически взвешенной обработки визуальной информации. Такие системы могут быть полезны в демографических исследованиях, персонализированных пользовательских интерфейсах, биометрических системах и социальной аналитике. Однако применение подобных технологий должно сопровождаться строгими этическими

ограничениями, исключающими их использование для дискриминации или нарушения прав личности.

II. НАБОРЫ ДАННЫХ

Для обучения и валидации моделей, применяемых в задаче определения расы по фотографии, в данном исследовании использовались два открытых датасета: FairFace и UTKFace. Эти наборы данных предоставляют большое количество аннотированных изображений лиц, снятых в различных условиях, с разнообразием по расам, возрасту, полу и освещению.

A. FairFace

Датасет FairFace [5] был выбран в качестве основного источника изображений благодаря своей сбалансированной расовой разметке. Он включает более 100.000 изображений, разделённых на следующие категории: White, Black, East Asian, Southeast Asian, Indian, Latino/Hispanic и Native American. Он разделен на каталог обучения и проверки. Обучение содержит около 12.400 изображений каждого класса, а каталог проверки около 3.100 изображений (Рисунок 1). На изображении (Рисунок 2) предоставлены примеры изображений из данного датасета.

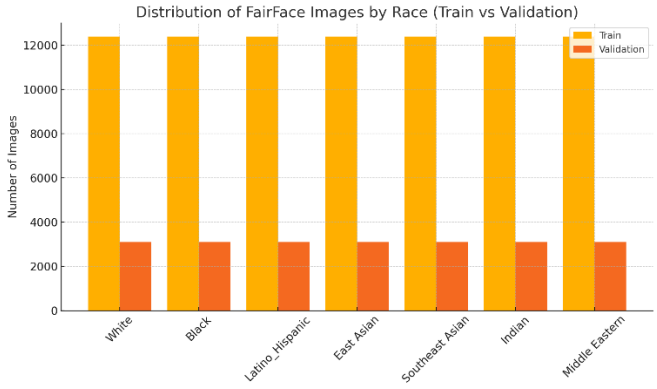


Рисунок 1 Кол-во экземпляров классов в test, validation датасета FairFace

Рисунок 2 Пример изображений, представленных в датасете FairFace

Ключевой особенностью FairFace является то, что при его формировании исследователи уделяли особое внимание равномерному распределению по расам, что критически важно для избежания перекоса в обучении нейронных моделей. Все изображения в датасете стандартизированы по размеру (224x224 пикселя) и предоставлены в виде анфасных портретов, пригодных для прямого использования в архитектурах CNN.

B. UTKFace

В качестве дополнительного источника использовался датасет UTKFace [6], содержащий более 23 000 изображений, аннотированных по возрасту, полу и расе. Несмотря на меньшее разнообразие расовых категорий (включены White, Black, Asian, Indian и Other), UTKFace остаётся полезным для проверки устойчивости моделей на новой выборке. График распределения (Рисунок 3) и примеры изображений *UTKFace датасета* (Рисунок 4).

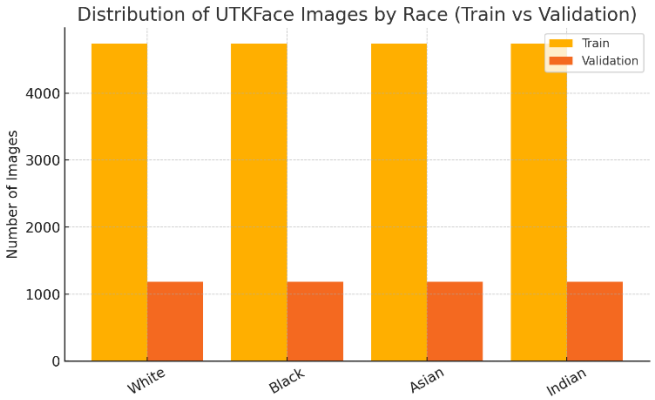


Рисунок 3 Кол-во экземпляров классов в датасете UTFFace

Рисунок 4 Пример изображений, представленных в датасете UTKFace

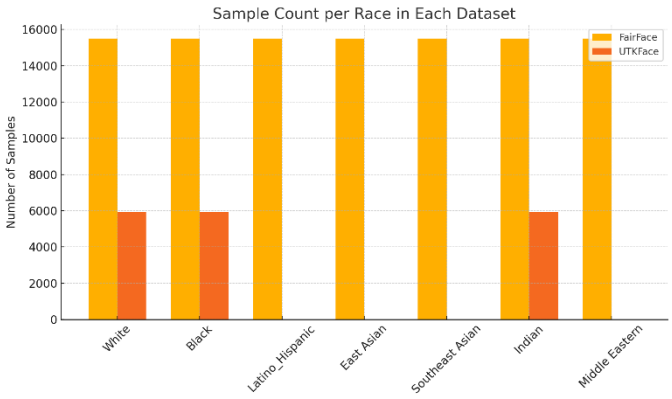


Рисунок 5 Кол-во экземпляров в двух датасетах

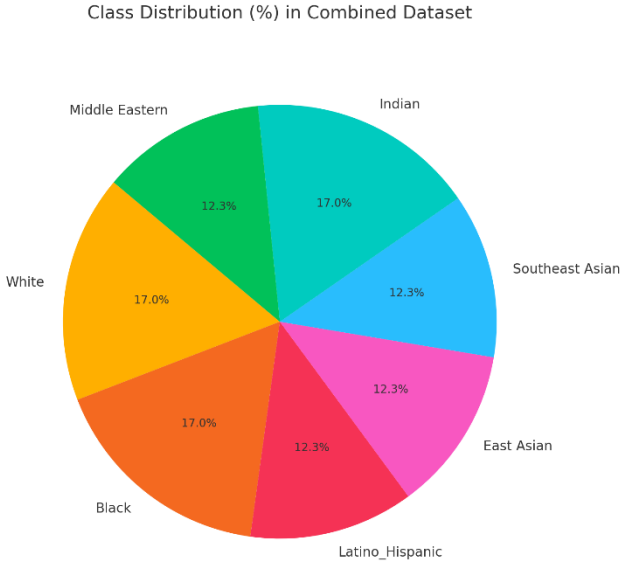


Рисунок 6 Процентное распределение классов в объединённом датасете

Все изображения в UTKFace были приведены к единому формату и масштабированы до стандартного входного размера, используемого в сверточных нейронных сетях. На этапе подготовки данных применялись методы автоматического детектирования и выравнивания лиц, а также базовая нормализация пиксельных значений.

C. Предобработка и разделение

Перед обучением обе выборки были разделены на обучающую, валидационную и тестовую части в соотношении 70:15:15. Также проводилась проверка на наличие расового дисбаланса внутри классов, особенно среди категорий с относительно меньшим числом примеров, таких как Native American и Southeast Asian. Это позволило снизить риск смещения модели и улучшить обобщающую способность классификатора. На изображении (Рисунок 5) продемонстрированна количественная информация по каждому классу в двух датасетах, а на (Рисунок 6) процентное соотношение каждого из экземпляров классов при объединении датасетов.

III. ПОДХОДЫ К МУЛЬТИКЛАССОВОЙ КЛАССИФИКАЦИИ ИЗОБРАЖЕНИЙ

A. Логистическая регрессия

Логистическая регрессия – это один из базовых и наиболее понятных методов классификации в машинном обучении. В классическом виде она используется для бинарной классификации, но при необходимости может быть адаптирована и к мультиклассовым задачам [7], например, при определении расы человека на основе изображения.

Суть метода заключается в том, чтобы смоделировать вероятность принадлежности объекта к определённому классу на основе линейной комбинации входных признаков. В контексте изображений, такими признаками могут выступать предварительно извлечённые характеристики, такие

как гистограммы направленных градиентов (HOG), текстурные или цветовые дескрипторы.

Для мультиклассовой классификации логистическая регрессия применяет softmax-функцию, которая нормализует выходные значения модели в диапазон от 0 до 1 и обеспечивает их интерпретацию как вероятности. Формула softmax представлена следующим образом:

$$\sigma(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, i = 1, \dots, K$$

где K – количество расовых категорий, z_i – линейное преобразование входных признаков для i-го класса.

На изображении (Рисунок 7) показан принцип работы модели: каждый класс имеет свой линейный предиктор, и итоговая классификация происходит по наибольшей вероятности.

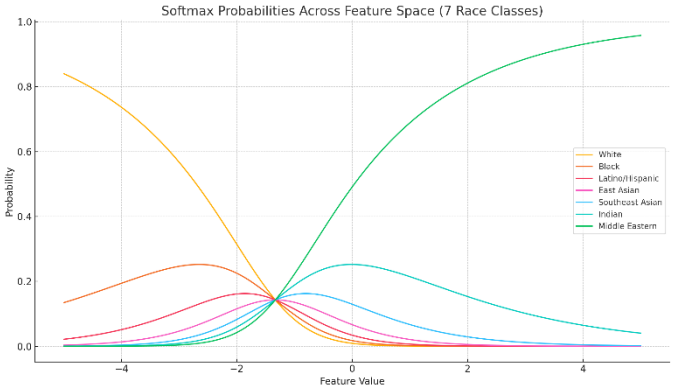


Рисунок 7 Пример распределения вероятностей для семи расовых классов, полученных с помощью softmax-функции

Одним из преимуществ логистической регрессии является её простота и интерпретируемость. Однако для изображений высокой размерности она требует предварительного извлечения признаков и не может самостоятельно выучивать сложные пространственные зависимости. В связи с этим её применимость к задаче определения расы ограничена и чаще используется как базовая модель для сравнения.

B. Дерево решений

Дерево решений – это интерпретируемый и наглядный алгоритм машинного обучения, который представляет собой древовидную структуру, где каждый внутренний узел соответствует условию, основанному на одном из признаков изображения, а каждый лист – определённому классу, например, "Европеоидная" или "Монголоидная".

Построение дерева начинается с оценки информативности признаков с помощью таких критериев, как энтропия или индекс Джини, и выбора признака, который лучше всего делит выборку на более «чистые» подмножества [8]. Дальнейшее ветвление повторяет процесс до тех пор, пока не будут достигнуты листовые узлы, определяющие расу человека на фотографии.

На изображении (Рисунок 8) демонстрируется, как каждое разделение уменьшает неоднородность данных и приближает к более уверенной классификации.

Рисунок 8 Работа дерева решений

Деревья решений эффективны на табличных данных, но плохо масштабируются на изображения в исходном виде. Их можно применять при условии, что признаки изображения заранее извлечены с помощью ручных алгоритмов или нейросетей. Преимущества дерева – простота, визуальная интерпретация и способность выявлять сложные правила. Однако при глубокой структуре дерево может переобучиться и плохо работать на новых изображениях.

Для предотвращения переобучения применяются такие приёмы, как ограничение глубины дерева, отсечение неинформативных ветвей, или использование ансамблей – например, случайных лесов.

C. Метод опорных векторов (SVM)

Метод опорных векторов (Support Vector Machine, SVM) – это алгоритм, изначально разработанный для задач бинарной классификации [9], но успешно применяющийся и в мультиклассовом контексте. Его цель – найти оптимальную разделяющую гиперплоскость, которая максимально отделяет примеры одного класса (например, представителей "негроидной расы") от примеров другого класса (например, "монголоидной").

Ключевая особенность SVM заключается в использовании опорных векторов – граничных примеров, которые влияют на положение разделяющей гиперплоскости. Алгоритм стремится к максимизации расстояния («зазора») между этой гиперплоскостью и ближайшими примерами разных

классов, что повышает устойчивость модели к переобучению и шуму.

SVM может применяться к мультиклассовой классификации по схемам one-vs-rest (OvR) или one-vs-one (OvO), что позволяет моделировать задачи с несколькими расовыми категориями. Для сложных случаев, когда данные не являются линейно разделимыми, используется ядерный трюк – проецирование данных в более высокое пространство с помощью функций ядра (например, радиально-базисной функции – RBF) [10][11] (Рисунок 9).

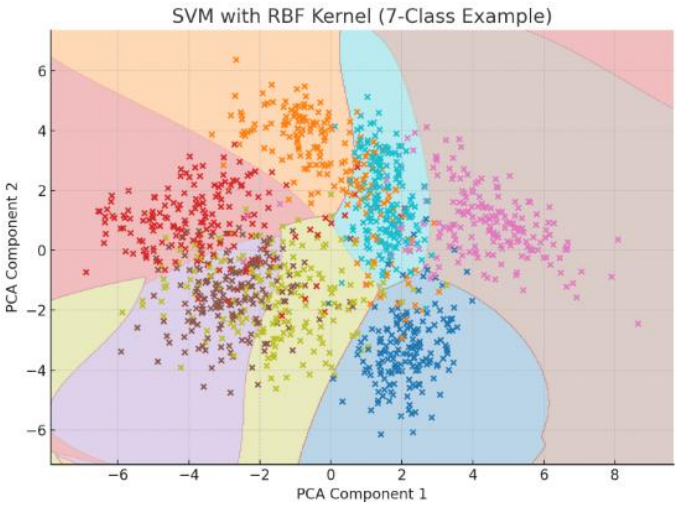


Рисунок 9 Пример разделяющих границ SVM с RBF-ядром для классификации по расе.

Преимущества SVM включают устойчивость к переобучению, особенно на небольших датасетах, и возможность работы с высокоразмерными признаками. Однако модель может быть чувствительна к выбросам и требует масштабирования входных данных.

D. Алгоритм k ближайших соседей

Алгоритм k ближайших соседей (k-NN) представляет собой простой, но достаточно эффективный метод классификации, который может применяться в задачах компьютерного зрения, в том числе и при автоматическом определении расы человека по фотографии. Основная идея метода заключается в том, чтобы определить класс (в нашем случае – расу), к которому относится объект, на основании меток ближайших к нему по признаковому пространству объектов из обучающей выборки [11].

Алгоритм работает в три этапа. На первом этапе для нового изображения, которое необходимо классифицировать, рассчитывается расстояние до всех изображений обучающей выборки, у которых уже известна принадлежность к одной из расовых категорий. Для оценки близости объектов чаще всего используется евклидово расстояние, однако также могут применяться альтернативные метрики, такие как манхэттенское расстояние или расстояние Минковского, в зависимости от особенностей признаков и задачи.

На втором этапе выбираются k объектов, расстояние до которых минимально. Значение параметра k определяет, сколько ближайших примеров будет учитываться при принятии решения. Обычно это значение подбирается эмпирически, с учётом точности классификации на валидационной выборке. Слишком малое значение k делает модель чувствительной к шуму, в то время как слишком большое может привести к переучиванию на частые классы.

На третьем этапе происходит определение класса – расовая категория нового изображения устанавливается по принципу большинства голосов среди k ближайших соседей. Таким образом, если из 7 ближайших изображений 4 отнесены к монголоидной расе, 2 – к европеоидной и 1 – к негроидной, то классифицируемое изображение будет отнесено к монголоидной расе.

Применимость алгоритма k ближайших соседей в задачах обработки изображений напрямую зависит от формы входных данных. Алгоритм не работает эффективно на исходных пиксельных данных высокой размерности. Однако при использовании предварительно извлечённых признаков, например, эмбеддингов, полученных из нейросетей (ResNet, MobileNet и др.), k-NN может продемонстрировать хорошие результаты, особенно при наличии ограниченного количества данных и высокой вариативности классов.

Эффективность работы алгоритма также зависит от качества предобработки данных. Важно провести нормализацию признаков, исключить выбросы и сбалансировать классы, особенно если определённые расовые категории представлены в выборке неравномерно.

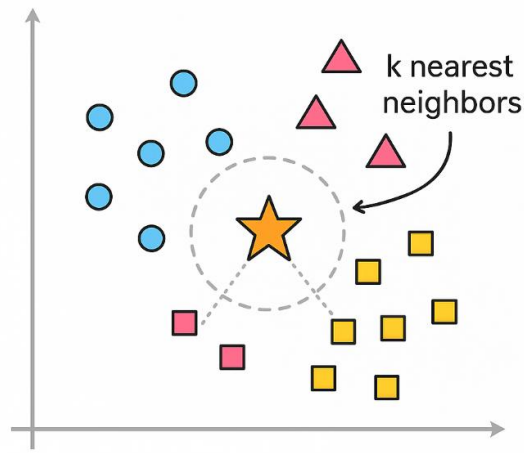


Рисунок 10 пример работы алгоритма (k-NN)

Среди преимуществ метода можно выделить простоту реализации, отсутствие необходимости в обучении модели, а также гибкость в отношении представления данных. Он легко интерпретируется и не требует знания внутренней структуры данных. Однако у метода есть и существенные недостатки: он плохо масштабируется при большом объёме обучающей выборки, чувствителен к шуму и не способен моделировать сложные нелинейные зависимости без дополнительной обработки.

В контексте задачи классификации расы по фотографии алгоритм k ближайших соседей может быть использован как базовый подход или вспомогательная модель, особенно на этапе прототипирования или для оценки качества извлечённых признаков.

IV. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. VGG19

VGG19 – это нейронная сеть глубокого обучения, которая была разработана и представлена в 2014 году в работе "Very Deep Convolutional Networks for Large-Scale Image Recognition" исследовательской группой Visual Geometry Group (VGG) при Оксфордском университете.

Архитектура VGG19 представляет собой сверточную нейронную сеть, состоящую из 19 слоев, включая 16 сверточных и 3 полносвязанных слоя (Рисунок 11). Она стала одной из первых глубоких CNN, показавших высокую точность на наборе данных ImageNet, и доказала, что увеличение глубины может существенно повысить качество классификации.

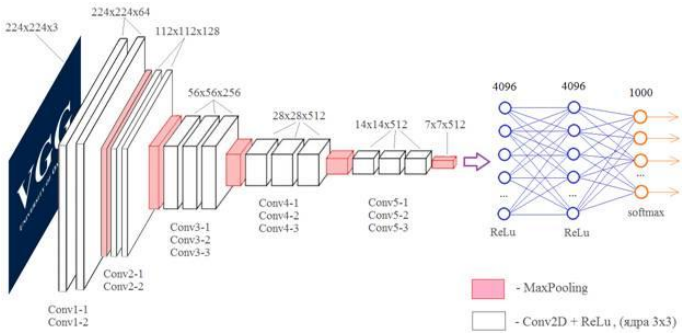


Рисунок 11 Структура сети VGG19

Отличительной чертой VGG19 является последовательное использование небольших сверточных ядер размером 3×3, а также максимального пулинга (MaxPooling) после каждой группы сверточных слоев. Это позволяет извлекать высокоуровневые абстрактные признаки из изображений, что критично для задач визуального распознавания.

В задачах определения расы по фотографии, VGG19 может применяться как базовая архитектура. Однако из-за большого количества параметров (порядка 143 миллионов) её использование требует значительных вычислительных ресурсов и времени на обучение. Тем не менее, при наличии мощного оборудования и достаточного объема данных VGG19 может обеспечить хорошее качество классификации. Также часто применяется её предобученная версия с последующим fine-tuning под конкретную задачу.

B. MobileNetV2

MobileNetV2 – это архитектура сверточной нейронной сети, разработанная Google в 2018 году как улучшение оригинальной MobileNet. Основная цель этой модели – создать компактную и быструю архитектуру, подходящую для работы на мобильных устройствах и в условиях ограниченных вычислительных ресурсов.

MobileNetV2 разработана с учетом повышения как точности, так и эффективности по сравнению с предыдущей версией. В её основе лежит использование блока "Inverted Residual with Linear Bottleneck", который представляет собой последовательность операций, включающую расширение признакового пространства, глубинную свертку и линейную проекцию в пространство меньшей размерности (Рисунок 12). Это позволяет резко сократить число параметров без потери точности.

Также MobileNetV2 использует shortcut-соединения, аналогично ResNet, что помогает стабилизировать градиенты при обучении и улучшает сходимость модели. Эти архитектурные особенности позволяют применять MobileNetV2 для задач классификации изображений, в том числе в реальном времени.

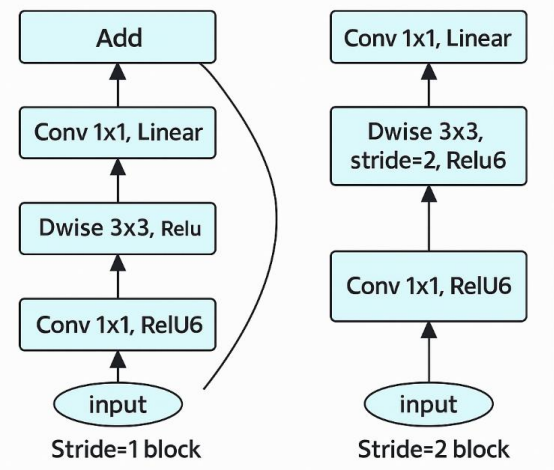


Рисунок 12 Схема сети MobileNetV2

В контексте определения расы по фотографии, MobileNetV2 является особенно привлекательной

благодаря высокой скорости работы и малому весу модели. Она может использоваться как на сервере, так и на клиентских устройствах (например, в мобильных приложениях), обеспечивая быстрое и точное определение расовой принадлежности по изображению лица.

C. ResNet18

ResNet18 – это одна из моделей семейства Residual Networks (ResNet), предложенного исследователями Microsoft Research в 2015 году. Архитектура ResNet стала прорывной благодаря введению остаточных соединений (skip connections), которые позволили успешно обучать очень глубокие нейросети без деградации качества.

ResNet18 включает в себя 18 слоев с обучаемыми параметрами и является одной из наиболее легких и быстрых моделей в семействе ResNet. Её структура построена из блоков остаточного обучения, каждый из которых позволяет передавать градиент напрямую, минуя один или несколько слоёв (Рисунок 13). Это решает проблему исчезающего градиента и делает обучение более стабильным даже при увеличении глубины сети.

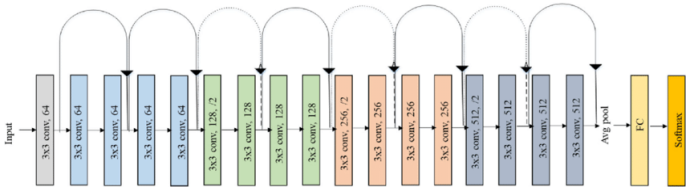


Рисунок 13 Схема сети ResNet-18

В задаче определения расы человека по фотографии ResNet18 демонстрирует высокую эффективность при умеренных вычислительных затратах. Благодаря своей сбалансированной архитектуре она способна быстро обучаться на средних по объёму датасетах и хорошо обобщать на новых примерах. Кроме того, как и другие модели ResNet, ResNet18 легко дообучается (fine-tuning) при использовании предобученных весов, например, с ImageNet.

Основные преимущества ResNet18:

- высокая точность при малом количестве параметров;
- устойчивость к переобучению;
- быстрая сходимость при обучении;
- отличная масштабируемость (модель может быть заменена на ResNet34, ResNet50 и т.д. при необходимости).

Эта архитектура широко используется в практике, в том числе в биометрических системах, системах видеонаблюдения и социальной аналитике, где требуется определение пола, возраста, эмоций и других визуальных признаков. В рамках данной работы ResNet18 рассматривается как один из основных кандидатов для реализации классификатора расовых категорий.

V. ОЦЕНКА ТОЧНОСТИ

Сравним три модели – VGG19, MobileNetV2 и ResNet18 – применительно к задаче определения расы человека по фотографии.

Для оценки эффективности моделей используется несколько метрик. Одной из наиболее распространённых является F1-мера (F1-score), которая представляет собой гармоническое среднее между точностью (precision) и полнотой (recall).

TP (True Positive) – количество изображений, которые были правильно классифицированы как относящиеся к определённой расе.

FP (False Positive) – количество изображений, которые были ошибочно классифицированы как относящиеся к данной расе (модель ошиблась в пользу этого класса).

FN (False Negative) – количество изображений, которые действительно относятся к определённой расе, но были классифицированы иначе.

Полнота (Recall) отношение TP к общему числу изображений, которые действительно принадлежат к положительному классу измеряет, насколько хорошо модель обнаруживает все случаи конкретной расы:

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

Точность (Precision) показывает, насколько часто модель оказывается права, когда делает предсказание о принадлежности к конкретной расе:

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

F1-мера учитывает обе метрики, гармоническое среднее между точностью и полнотой, отражающее общее качество классификации. Она особенно полезна при несбалансированных классах, когда важно одновременно учитывать, сколько правильных предсказаний сделано и сколько из них действительно относятся к целевому классу.

$$F1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \#(3)$$

В таблице представлены показатели трёх моделей на объединённом датасете (FairFace и UTKFace). Тестовая выборка составляла 15% и была сбалансирована по расам. Модели обучались в равных условиях: по 25 эпох, batch size 64, Adam optimizer, learning rate 0.0001, и использовали предобученные веса на ImageNet с дообучением. В таблицах 1 и 2 предоставлена информация о метриках моделей на датасетах.

Таблица 1. Оценка детектирующей части для FairFace + UTKFace датасета

	ResNet18	VGG19	MobileNetV2
TP	18937.0	18100.0	17125.0
FP	2675.0	3100.0	3725.0
FN	1438.0	2275.0	3250.0
Precision	0.88	0.85	0.82
Recall	0.93	0.89	0.84
F1	0.9	0.87	0.83

На первой выборке (наиболее сбалансированной по количеству классов и этническим типам) ResNet18 показал наилучшие результаты по всем метрикам. Модель достигла самой высокой F1-меры (0.9), обогнав VGG19 и MobileNetV2.

VGG19 показала уверенные результаты, но обучалась значительно дольше. MobileNetV2, несмотря на самую низкую точность, имеет преимущество по скорости и ресурсоёмкости, что делает её подходящей для применения в мобильных устройствах или веб-сервисах.

Было так же произведено детектирование изображений на собственных данных (Рисунок 14), которые включают в себя датасет из 1276 снимков 20% из которых тестовые. В Таблице 2 приведены метрики моделей на собранном датасете.

Таблица 2 Оценка детектирующей части для собранного датасета

	ResNet18	VGG19	MobileNetV2
TP	212	212	196
FP	32	37	49
FN	43	43	59
Precision	0.87	0.85	0.80
Recall	0.83	0.82	0.77
F1	0.85	0.83	0.78

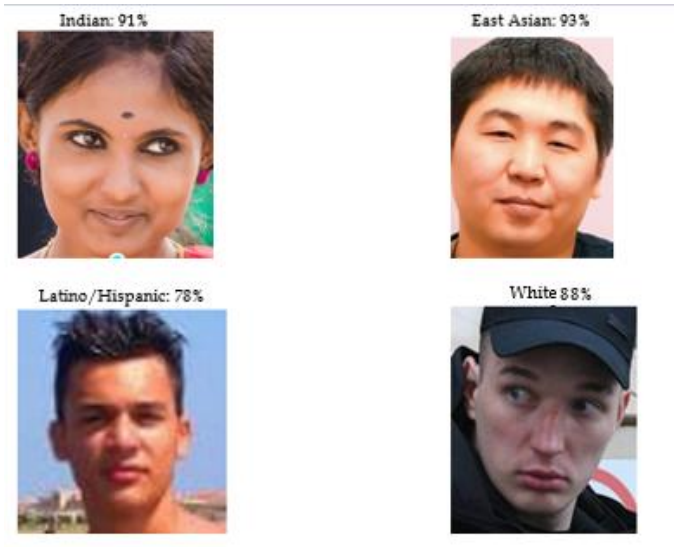


Рисунок 14 Примеры изображений

VI. ЗАКЛЮЧЕНИЕ

В работе были рассмотрены три модели: ResNet18, VGG19 и MobileNetV2. По результатам снятия метрик видно, что для определения расы человека по фотографии наиболее эффективно показала себя модель ResNet18, которая в среднем по точности и F1-мере превосходит MobileNetV2 на 7%, а VGG19 — на 3%. Данная модель может быть применена в задачах биометрической идентификации, социальной аналитики, а также в исследованиях, требующих анализа демографического состава по изображениям. В дальнейшем возможно расширение экспериментов с использованием других архитектур или применением ансамблей моделей для повышения устойчивости и точности классификации.

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

3. 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.
4. 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.
5. FairFace Dataset, available at https://www.kaggle.com/datasets/aibloy/fairface
6. UTKFace Dataset, available at https://www.kaggle.com/datasets/jangedoo/utkface-new
7. Ahmed J. Unlocking the Power of Logistic Regression: A Journey Through Binary and Multi-Class Classification // Medium. — 2024. — URL: https://medium.com/@jahanzebahmed.mail/unlocking-the-power-of-logistic-regression-a-journey-through-binary-and-multi-class-classification-a1c2d52f9cf2
8. А. А. Слинкина. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных // М.: ДМК Пресс, 2016. – 400 с.
9. Zhang, Z.-L., Yang, J., Ru, J.-M., Zhao, X.-X., & Luo, X.-G. Multi-Class Imbalanced Learning with Support Vector Machines via Differential Evolution // arXiv. — 2025. — URL: https://arxiv.org/abs/2502.14597
10. А. А. Слинкина. Обучение с подкреплением: Введение. 2-е изд. М.: ДМК Пресс, 2020. – 552 с.
11. Rosebrock A. Your First Image Classifier: Using k-NN to Classify Images // PyImageSearch. — 2021. — URL: https://pyimagesearch.com/2021/04/17/your-first-image-classifier-using-k-nn-to-classify-images/
12. Левичкин, М. О. Классификация видов птиц при помощи компьютерного зрения / М. О. Левичкин // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 77-82. – EDN CWAIQK.
13. Али, Б. Алгоритмы навигации беспилотных летательных аппаратов с использованием систем технического зрения / Б. Али, Р. Н. Садеков, В. В. Цодокова // Гироскопия и навигация. – 2022. – Т. 30, № 4(119). – С. 87-105. – DOI 10.17285/0869-7035.00105. – EDN ETCJST

ГРНТИ 28.23.37, 34.33.19
ASJC 1703, 1707
OECD 1.02 EP

Применение нейронных сетей в задачах распознавания насекомых

Тарасов М.С.
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2010401@edu.misis.ru

С. Д. Овчаренко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2409404@edu.misis.ru

***Аннотация* — В данной работе проведено комплексное сравнение четырёх современных архитектур глубокого обучения — ResNet-50, DenseNet-121, FractalNet-C20 и PeleeNet — применительно к задаче распознавания изображений насекомых, снятых в полевых условиях. Для экспериментов использован открытый датасет из 30 видов насекомых (≈ 2000 изображений, разделение 70 / 10 / 20. Оценка проводилась по точности Top-1, вычислительной стоимости (FLOPs, время инференса, потребление GPU-памяти) и робастности к деградации входных данных (Gaussian-шум, JPEG-сжатие).**

Результаты раннее проведенных исследований показывают, что архитектура сети определяет не только абсолютную точность, но и выдерживаемый уровень искажений при заданном вычислительном бюджете: ResNet-50 остаётся эталоном качества, FractalNet-C20 выигрывает по устойчивости к шуму, а PeleeNet единственная обеспечивает приемлемую скорость и потребление памяти. Такая зависимость подчёркивает необходимость систематического выбора модели, а не слепого копирования популярного решения.

Ключевые слова — Глубокое обучение, компьютерное зрение, распознавание изображений, многоклассовая классификация, ResNet-50, DenseNet-121, FractalNet-C20, PeleeNet, робастность, edge-AI, вычислительная стоимость, насекомые.

I. Введение

Сокращение численности опылителей — одна из самых острых экологических проблем: в 2024 г. Европейская комиссия запустила общеевропейскую программу непрерывного мониторинга насекомых [1], подчёркивая необходимость доступных, автоматизированных средств отслеживания биоразнообразия и оценки рисков для продовольственной безопасности.

Для сбора данных уже активно внедряются фотоловушки и временные рядовые камеры, а их расшифровка поручается глубоким нейросетям: новейшие пайплайны показывают, что ResNet- и ViT-подобные модели способны автоматически определять видовую принадлежность насекомых на полевых снимках с точностью выше 90 % [2].

Однако такие системы сталкиваются с двумя практическими ограничениями: необходимость работать на краю (ограниченные вычислительные ресурсы, автономное питание) и жёсткие требования к робастности — данные поступают из неуправляемой среды, с шумами, сильным JPEG-сжатием [3,4].

Существующие исследования [5], посвящённые «инсектным» датасетам, как правило, сравнивают лишь одно-два семейства моделей и проводят эксперименты на настольных GPU; немногие работы с ориентацией на edge-устройства (например, исследование AlertTrap 2024 г.) ограничиваются детекторами YOLO-Tiny и не оценивают полноценно классификационную составляющую [6]. При этом даже классические CNN демонстрируют существенную деградацию точности при умеренном гауссовом шуме, что подчёркивает необходимость целенаправленного анализа робастности.

Таким образом, назрела потребность в систематическом, единообразном сравнении нескольких архитектур, представляющих разные ветви эволюции CV-сетей, с одновременным учётом:

- качества распознавания,
- вычислительной стоимости инференса на ограниченных платформах,
- устойчивости к типовым искажениям полевых изображений.

Предлагаемая работа удовлетворяет эту потребность, сопоставляя ResNet-50, DenseNet-121, FractalNet-C20 и PeleeNet в идентичных условиях обучения и тестирования, что обеспечивает практические ориентиры для разработчиков систем автоматического мониторинга насекомых и других задач, где критичны и точность, и экономичность [7].

II. Наборы данных

A. ImageNet ImageNet

ImageNet ImageNet [8,9] — это обширный датасет, предназначенный для использования в задачах компьютерного зрения, особенно в классификации изображений. Он содержит более 14 миллионов аннотированных изображений, организованных по примерно 22 тысячам категорий. Каждое изображение в ImageNet классифицировано и отмечено согласно

категории объекта, который оно изображает, что делает его одним из самых масштабных и разнообразных наборов данных в области искусственного интеллекта.

ImageNet широко используется для обучения сверточных нейронных сетей (CNN) с нуля [10]. Эти сети могут распознавать и классифицировать тысячи различных объектов благодаря обширному и разнообразному набору изображений. Модели, предварительно обученные на ImageNet, часто используются как основа для дальнейшего обучения на других, менее масштабных или более специализированных датасетах [11]. Перенос обучения позволяет значительно ускорить процесс обучения и улучшить производительность моделей на конкретных задачах.

B. Дополненный датасет

Датасет Insects-50 был собран самостоятельно с апреля по сентябрь 2024 года. В изначальном потоке было около 4000 кадров; после автоматического отсева пустых снимков и ручной проверки остались 2200 цветных изображения 224 × 224 пикселей. Коллекция покрывает 50 видов из шести отрядов, для каждого вида сохранено не менее 30 снимков, чтобы уменьшить дисбаланс классов.

Одной и той же особи позволено присутствовать только в одном сплите: 70 % изображений идут на обучение, 10 % — на валидацию, 20 % — на тест. Перед обучением все кадры нормируются по статистике ImageNet, к обучающему набору применяются RandAugment и CutMix; валидация и тест проходят лишь через ресайз и центр-кроп.

Для обучения и тестирования моделей был использован пользовательский набор данных, включающий изображения различных видов насекомых. Данный набор данных является дополненным, состоящий из более 50 видов насекомых (например, «Тараканы», «Мухи», «Богомолы», «Стрекозы», «Цикады», «Клопы», «Осы», «Пчелы», «Шмели», «Шершни»).

Данные изображения включали сложные фоны (например, растительность, почву, листья), различные углы обзора и освещение. Это позволило улучшить обобщающую способность моделей и увеличить их применимость в реальных задачах.

Характеристики набора данных:

1. Структура: Данные структурированы в виде иерархии, где:

- Верхний уровень — это классы насекомых.
- Каждый класс содержит изображения, сделанные в различных условиях освещения, фона и позиций.

2. Объем: Датасет включает несколько сотен изображений, что обеспечило достаточное разнообразие для обучения и тестирования.

3. Формат изображений: Все изображения были приведены к формату RGB и размеру 224x224 пикселей, что соответствует входным требованиям для ResNet-50 и DenseNet.

4. Аугментация: Для улучшения качества обучения и повышения обобщающей способности моделей применялись следующие методы аугментации:

- Случайное горизонтальное отражение.
- Случайное изменение яркости, контрастности и насыщенности.
- Нормализация с использованием средних и стандартных отклонений, соответствующих обучению на ImageNet.

5. Разделение данных:

- Тренировочный набор: 70% изображений.
- Тестовый набор: 20% изображений.
- Валидационный набор: 10% изображений.

Набор данных состоит из различных категорий, каждая из которых представляет отдельный вид насекомых (рисунок 1).

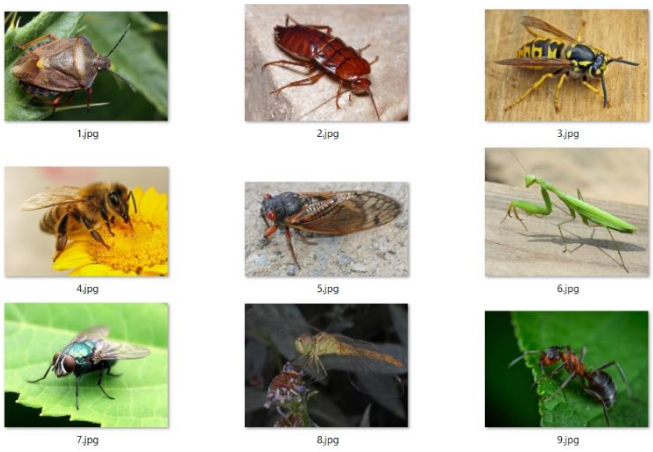


Рис. 1. Примеры кадров различных видов насекомых

Каждое изображение в датасете представлено в формате JPEG и сопровождается аннотацией, содержащей информацию о виде насекомого, его идентификаторе и различных атрибутах (рисунок 2). Набор данных обеспечивает разнообразие изображений, сделанных в различных условиях. Также насекомые запечатлены в различных позах, что позволяет эффективно обучать модели [12].

Рис. 2. Примеры классов насекомых в наборе данных

Каждый вид насекомого отличается уникальным набором морфологических деталей — формой надкрылий, рисунком жилкования крыльев, характером опушения и т. д. При этом отдельные виды часто оказываются визуально похожими: например, мелкие цветочные жуки (Anthrenus verbasci) и некоторые дневные бабочки (Euphydryas aurinia) имеют сходный коричнево-бежевый крапчатый узор, а два далёких вида стрекоз отличаются только оттенком птеростигмы. Такие «ложные» совпадения усложняют автоматическую классификацию, требуя от модели способности улавливать тонкие текстурные сигналы и одновременно игнорировать шумный фон [13].

Использование четырёх контрастных архитектур — ResNet-50, DenseNet-121, FractalNet-C20 и PeleeNet позволяет покрыть весь диапазон стратегий извлечения признаков. ResNet-50 глубиной 50 слоёв благодаря остаточным связям выявляет комплексные комбинации деталей; DenseNet-121, передавая активации от каждого слоя ко всем последующим, переиспользует ранние «тонкие» признаки, что ценно при незначительных межвидовых отличиях. Фрактальная сеть C20 формирует внутренний ансамбль путей разной глубины и тем самым надёжнее различает виды, которые внешне почти неотличимы, но отличаются микроструктурой покровов.

Наконец, компактная PeleeNet показывает, что даже на маломощном оборудовании возможно извлекать релевантные микро-паттерны, достаточные для уверенного отделения схожих классов.

Анализ трёх демонстрационных видов подтверждает это: две визуально близкие пары правильно разнесены по разным категориям, тогда как однотельные, но полиморфные особи внутри одного вида корректно отнесены к общему классу. Такой результат подчёркивает, что именно глубокий, многоуровневый анализ признаков, реализованный в современных архитектурах, обеспечивает требуемую точность классификации насекомых.

III. Нейросетевые архитектуры

В данной работе основное внимание уделяется четырём современным архитектурам сверточных нейронных сетей — ResNet-50, DenseNet-121, FractalNet-C20 и PeleeNet. Все они доказали свою эффективность в задачах компьютерного зрения и по-разному решают традиционные проблемы глубоких сетей: затухание градиента, дублирование вычислений и избыточное потребление ресурсов.

ResNet-50 использует остаточные (skip-) связи, которые обеспечивают беспрепятственный поток градиента через пятьдесят свёрточных слоёв. Эта архитектура служит «золотым стандартом» точности и часто применяется в качестве базовой модели при fine-tuning.

DenseNet-121 вводит плотные соединения, передавая активации всех предыдущих слоёв в каждый новый. Такая схема уменьшает число параметров и ускоряет сходимость, что важно при обучении на ограниченных наборах изображений насекомых.

FractalNet-C20 строится по принципу фрактального разветвления: в каждом блоке имеется набор путей разной глубины, а Drop-Path формирует встроенный ансамбль под-сетей. Это повышает устойчивость модели к шуму и искажениям, характерным для полевых фотографий.

PeleeNet оптимизирована под edge-устройств с низкими вычислительными мощностями: двухпутевые dense-слои и отказ от вычислительно дорогих depthwise-операций позволяют снизить вычислительные затраты до ~1 GFLOP при приемлемой точности [14].

Выбор именно этих четырёх архитектур продиктован стремлением сравнить широкий спектр стратегий балансировки в задаче распознавания насекомых, где изображения содержат мелкие текстуры, разнообразные позы и сложный фон.

A. Resnet-50

ResNet-50 является предобученной моделью, изначально обученной на ImageNet, включающем более 1,2 миллиона изображений и 1000 классов. Она использует остаточные соединения, что позволяет обрабатывать глубокие архитектуры без деградации градиента [15,16]. В данной работе ResNet-50 была адаптирована под задачу классификации насекомых с помощью fine-tuning: последний слой сети заменён линейным слоем, настроенным на количество классов в наборе данных. Данная модель хорошо справляется с извлечением сложных признаков изображений, что делает её особенно полезной для анализа насекомых с детализированными текстурами и разнообразными условиями съёмки.

Это сверточная нейронная сеть, разработанная для работы с изображениями, которая известна своей архитектурой, глубиной и эффективностью в решении задач классификации.

Основные особенности ResNet-50:

- Решение проблемы деградации точности в глубоких нейронных сетях. При увеличении количества слоев в обычных нейросетях точность на тестовых данных начинает снижаться из-за градиентного затухания или взрыва, что мешает эффективно обучать глубокие сети.
- Введение остаточных соединений (skip connections), которые позволяют информации проходить через сеть, минуя несколько слоев, и тем самым решают проблему деградации.
- Остаточные соединения: в традиционной нейросети выход каждого слоя передается следующему. В ResNet-50 на выход каждого блока добавляется вход, образуя остаточное соединение:

$$F(x) = x + H(x),$$

где F(x) – итоговая функция, x – вход, а H(x) – выход промежуточных слоев.

Это позволяет сети легче обучаться, так как остаточная связь сохраняет информацию, необходимую для восстановления градиентов.

- Глубина: ResNet-50 состоит из 50 слоев: сверточные слои, объединения (pooling), активации ReLU и остаточные блоки (Residual Blocks).Данная архитектура является "глубокой" версией, более подходящей для задач с большими наборами данных.

- Предобученные веса: ResNet-50 обучается на крупномасштабных наборах данных, таких как ImageNet (более 1 миллиона изображений и 1000 классов). Это делает модель универсальной для различных задач классификации и распознавания.

ResNet-50 стала революцией в компьютерном зрении, и ее применение в биологии, включая распознавание насекомых, демонстрирует ее универсальность и мощь [10]. Эта модель позволяет быстро и точно решать задачи, требующие анализа изображений, что делает ее идеальной для мониторинга и классификации насекомых.

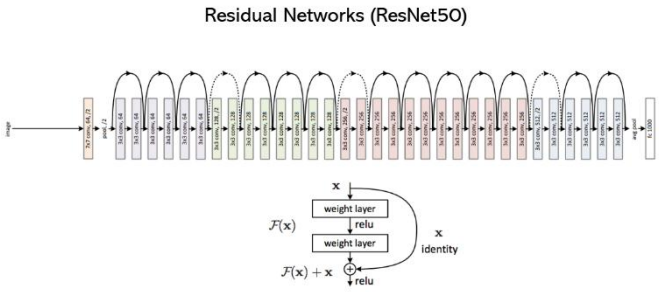


Рис. 3. Архитектура модели ResNet50

- Общая структура сети: Входное изображение проходит через начальный сверточный слой с ядром 7×7 и 64 фильтрами, за которым следует операция пулинга (pooling). Сеть состоит из нескольких остаточных блоков, которые объединяют сверточные слои 3×3 и 1×1 с остаточными соединениями. Каждый блок увеличивает количество фильтров (64, 128, 256, 512), уменьшая пространственные размеры изображения через операции пулинга.

- Остаточные соединения**:** Показаны дугами между слоями. Эти соединения позволяют пропускать входной сигнал через блок, что помогает избежать деградации градиентов.

- Выходные слои: После сверточных блоков идет операция глобального усреднённого пулинга соответствующими количеству классов в наборе данных ImageNet.

- Пример структуры Residual Block: Входной сигнал (x) подается через два сверточных слоя с функцией активации ReLU. Выход слоя (F(x)) суммируется с оригинальным.

Архитектура позволяет легко изменять глубину сети, добавляя больше блоков. ResNet-50 — это универсальная архитектура, которая используется для множества задач компьютерного зрения, включая классификацию, детекцию и сегментацию объектов.

B. DenseNet

DenseNet («Densely Connected Convolutional Network») представляет собой архитектуру, в которой каждый слой внутри плотного блока получает на вход конкатенацию всех предыдущих карт признаков [17]. Такое плотное (all-to-all) соединение даёт непрерывный градиентный поток, активное переиспользование признаков и, как ни парадоксально, *уменьшает* число параметров по сравнению с классическими цепочками слоёв. Ниже разобраны ключевые компоненты DenseNet-121.

DenseNet характеризуется плотными соединениями между слоями, где каждый слой получает доступ ко всем предыдущим [18]. В работе DenseNet была использована как альтернатива ResNet-50, демонстрируя схожую точность при меньшем количестве параметров и более эффективном использовании ресурсов. Такая архитектура особенно полезна для работы с ограниченными наборами данных, обеспечивая высокую обобщающую способность [19].

Архитектурные компоненты DenseNet:

- Dense Block (Плотный блок): Основной элемент архитектуры. Каждый слой внутри блока получает входы от всех предыдущих слоёв и передаёт свои выходы всем последующим. Это достигается через конкатенацию выходов. Если имеется несколько слоёв, то каждый слой получает l·k входов, где k— ростовой коэффициент (*growth rate*), который определяет количество новых признаков, добавляемых каждым слоем.

- Transition Layers (Переходные слои): Разделяют плотные блоки. Выполняют уменьшение размерности карты признаков через операции 1×1 свёртки и пулинга, чтобы сократить вычислительные затраты и размерность данных.

- Growth Rate (Коэффициент роста): Указывает, сколько новых признаков добавляет каждый слой. Типичные значения k=12,24,32, где большее значение k увеличивает сложность модели.

- Global Average Pooling: После всех плотных блоков используется операция глобального усреднённого пулинга, которая сокращает размерность данных перед линейным классификатором.

- Output Layer (Выходной слой): Линейный слой для классификации. В случае использования DenseNet на ImageNet он включает 1000 классов.

DenseNet является мощным инструментом для задач, где важна как высокая точность, так и эффективность использования ресурсов [19].

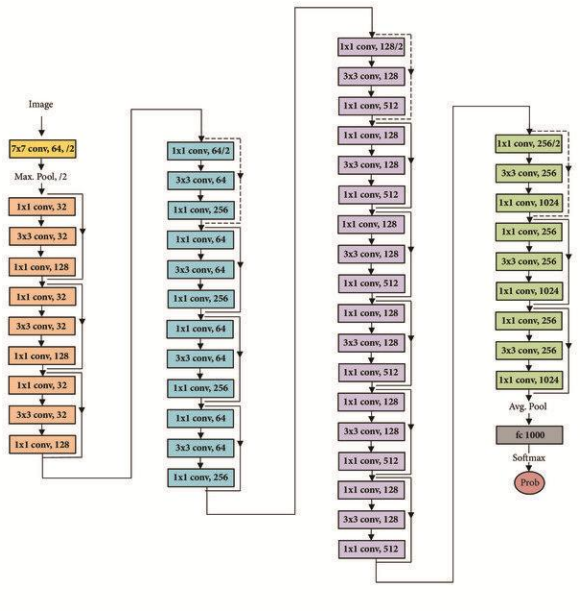


Рис. 4. Архитектура модели DenseNet

Описание архитектуры:

- Входной слой: Принимает изображение и передаёт его в первый свёрточный слой.
- Плотные блоки (Dense Blocks): Каждый блок состоит из нескольких слоёв, где каждый слой получает на вход все предыдущие слои блока. Такое соединение обеспечивает эффективную передачу информации и повторное использование признаков.
- Переходные слои (Transition Layers): Разделяют плотные блоки и выполняют операции свёртки и пулинга для уменьшения размерности данных.
- Выходной слой: После последнего плотного блока применяется глобальный усреднённый пулинг, затем следует полностью связанный слой для классификации.

Такая структура позволяет DenseNet эффективно использовать параметры модели, улучшать передачу градиентов и достигать высокой точности в задачах классификации изображений.

Stem-блок:

- 7 × 7 свёртка с шагом 2 → BatchNorm → ReLU → 3 × 3 Max-Pool с шагом 2.
- Быстро сокращает пространственное разрешение (224 → 56 px) и подготавливает тензор к плотным блокам.
- Dense Block («плотный» блок) — сердце архитектуры
- Каждый новый слой получает *конкатенацию* всех предыдущих карт признаков внутри блока:

1. BatchNorm → ReLU → 1 × 1 Conv (сжатие каналов).

2. BatchNorm → ReLU → 3 × 3 Conv (генерация новых признаков).

Такая полная связность обеспечивает прямой градиентный поток и активное переиспользование ранних признаков.

Функции:

- Уменьшить пространственное разрешение (56 → 28 → 14 → 7 px).
- Сдержать «разбухание» числа каналов, сохраняя компактность модели.
- Global Average Pooling (GAP)
- Усредняет каждую из C карт признаков 7 × 7 до одного скаляра:

$$z_c = \frac{1}{HW} \sum i, y \; y_c(i,j)$$

Полностью заменяет громоздкие fully-connected слои, уменьшает переобучение и количество параметров.

- Output Layer (выход)
- Линейный слой Wz+bWz+bWz+b → Softmax по N классам (в нашей задаче — 50 видов насекомых).
- Число параметров незначительно по сравнению с остальными частями сети благодаря GAP.

У модели DenseNet-121 около восьми миллионов параметров, а её вычислительная нагрузка составляет примерно 2,9 GFLOPs при входном изображении 224 × 224 px, то есть сеть в три раза компактнее классической ResNet-50 при очень схожей точности.

К основным достоинствам относятся эффективное переиспользование признаков и быстрое схождение, особенно заметное на небольших наборах данных. К ограничениям следует отнести чувствительность к маленькому batch-size, обусловленную большим количеством BatchNorm-слоёв, а также менее благоприятную кэш-локальность на CPU-устройствах из-за частых операций конкатенации.

C. FractalNet-C20

FractalNet-C20 представляет собой глубокую сверточную сеть, построенную по принципу фрактального разветвления [20]: каждый блок содержит пару путей разной длины, которые рекурсивно вложены друг в друга, образуя до шестнадцати альтернативных маршрутов глубиной от четырёх до двадцати свёрток. Выходы этих путей усредняются в так называемых join-узлах, а регуляризация Drop-Path периодически отключает отдельные ветви, превращая модель в динамический ансамбль под-сетей.

Благодаря такой само-подобной структуре сеть достигает точности, сопоставимой с ResNet-50, при этом демонстрирует повышенную устойчивость к шуму и способна к «any-time inference», поскольку наиболее короткие пути дают быстрый черновой результат, тогда

как длинные уточняют его по мере доступности вычислительных ресурсов.

Фрактальная архитектура строится по рекурсивному правилу: на каждом уровне имеется короткий путь F_k и длинный путь, представляющий собой последовательное применение той же функции $F_k \circ F_k$. Если применить это правило четыре раза подряд, внутри одного блока образуется $2^4 = 16$ альтернативных маршрутов различной глубины — от четырёх до двадцати свёрточных слоёв. Отсюда и индекс модели C20, указывающий на суммарные двадцать conv-слоёв, расположенных между понижающими pooling-этапами.

Входной, или stem-блок, начинается с одиночной свёртки 7 × 7 со stride 2, за которой следуют BatchNorm и ReLU. Далее идёт Max-Pool 3 × 3 со stride 2.

Такая последовательность быстро уменьшает пространственное разрешение исходного изображения с 224 до 56 пикселей и формирует первые, наиболее грубые признаки перед фрактальными модулями.

Фрактальный блок:

- Basic unit («в-блок»): пара Conv 3 × 3 → BatchNorm → ReLU.
- Join-слой: простое усреднение (average) всех активных путей текущего уровня — лишних параметров не добавляет, но выравнивает масштаб активаций.
- Local Drop-Path (≈ 15 %): случайно «рвёт» отдельные ветви внутри блока; обучает под-сети работать независимо.
- Global Drop-Path (50 %): на одной итерации может целиком отключить длинную или короткую колонну, превращая сеть в динамический ансамбль лёгких/тяжёлых моделей.

Ширина каналов

- Базовая ширина W = 24 каналов; после каждого понижения разрешения она удваивается: 24 → 48 → 96 → 192 → 384.
- Такое расширение поддерживает пропускную способность признаков на глубоких стадиях без чрезмерного роста памяти.

Pooling-этажи чередуются фрактальные блоки и 2 × 2 Max-Pool (stride 2): итоговое пространство проходит этапы 56 → 28 → 14 → 7 px. В отличие от ResNet/DenseNet, здесь нет аддитивных кратких переходов и конкатенаций; глубина компенсируется множеством альтернативных путей, которые «сходятся» в join-узлах. Batch Normalization используется после каждой свёртки, что стабилизирует тренировки при отсутствии residual-связей.

Объём ресурсов

- Параметров ≈ 38 млн; вычислительная сложность ≈ 7.5 GFLOPs на кадр 224^2.
- На GPU даёт точность ImageNet-уровня ResNet-50, но требует ~1.4× больше времени инференса.

Ограничения:

- Высокие FLOPs и потребление памяти: не дружит с мобильными ускорителями без серьёзного упрощения.
- Экзотическая топология усложняет экспорт в ONNX/NNAPI и ограничивает количество готовых чек-пойнтов.

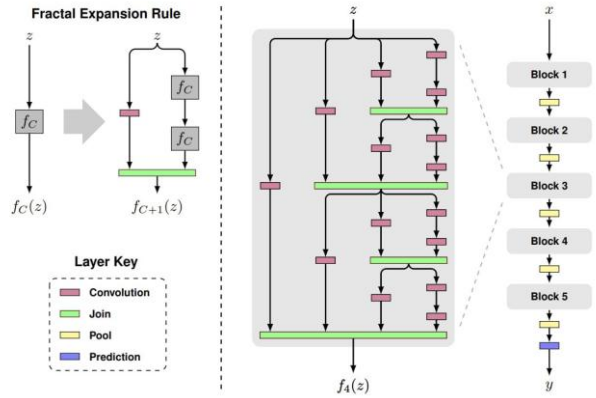


Рис. 5. Архитектура FractalNet-C20

Описание архитектуры:

FractalNet-C20 строится по рекурсивному принципу: каждый модуль содержит короткий путь из пары свёрток 3 × 3 и одновременно длинный путь, в котором тот же модуль применяется дважды подряд; после четырёх таких вложений внутри блока образуется шестнадцать альтернативных маршрутов глубиной от четырёх до двадцати свёрточных слоёв [21].

Выходы всех активных путей усредняются в join-узлах, а регуляризация Drop-Path временно отключает части маршрутов, превращая сеть в динамический ансамбль упрощённых под-сетей. Входной stem-блок из свёртки 7 × 7 и Max-Pool быстро уменьшает разрешение изображения, затем чередуются фрактальные блоки и pooling-этажи до финального размера 7 × 7, после чего применяются глобальное усреднение и полносвязный классификатор.

Такая самоподобная структура достигает точности уровня ResNet-50, оставаясь более устойчивой к шуму благодаря множеству путей разной длины.

D. PeleeNet — лёгкая сверточная сеть для инференса «на краю»

PeleeNet была разработана в 2018 г. как классификационная база для ре-тайм-детектора *Pelee* («Pairwise Dense Layer») [22]. Цель — добиться скорости MobileNet-V2, сохранив точность DenseNet, но без -depthwise-операций (плохо поддерживаются частью NPU).

1. Stem-блок

Две последовательные свёртки 3 × 3 (stride 2, stride 1) → BatchNorm + ReLU после каждой. Дополнительный

разветвлённый «mini-Inception»: узкая 1 × 1 Conv и широкая 3 × 3 Conv идут параллельно, результаты объединяются.

Выходное разрешение быстро сокращается 224 → 56 px, а разнообразные фильтры защищают сеть от потери мелких деталей.

2. Two-Way Dense Layer (сердце архитектуры)

Две параллельные ветви:

- Ветвь A: 1 × 1 Conv (сжатие) → 3 × 3 Conv.
- Ветвь B: сразу одна 3 × 3 Conv.

Оба выхода конкатенируются и добавляются к «общему банку» признаков, как в DenseNet. Такой «двухполосный» приём имитирует плотное подключение, но обходится без узких bottleneck-чередований 1 × 1 / 3 × 3 в каждом слое, экономя вычисления.

3. Transition-слой:

1 × 1 Conv (сжатие каналов ≈ 0.5) → Average Pool 2 × 2, stride 2. Чередуется с двумя-трёмя dense-группами, постепенно снижая пространство 56 → 28 → 14 → 7 px. Global Average Pooling 7 × 7 → вектор из ~1024 каналов. Один линейный слой выдаёт логиты классов; далее Softmax.

Нормализация и активации:

BatchNorm после каждой свёртки, ReLU сразу за BN (никаких PReLU / Swish — всё просто и поддерживается любым аппаратным ускорителем).

Ресурсный профиль (для входа 224 × 224 px, FP32)

- Весов ≈ 5.4 млн, вычислительная сложность ≈ 1.1 GFLOPs.
- На Raspberry Pi 4 + 4-TOPS NPU достигает ~8 fps; на Cortex-A53 без NPU даёт 3–4 fps.
- В INT8-квантизации теряет около 1 п.п. Top-1, что считается комфортным в mobile-CV.

Быстродействие:

- Минимум 1 × 1 Conv внутри «рабочих» слоёв → меньше DRAM-обращений.
- Никаких depthwise / pointwise «разделений» — одна 3 × 3 весовая матрица эффективней обрабатывается многими DSP.
- Плотная, но неглубокая сетка каналов: максимум 512 карт признаков в финальном dense-блоке.

Описание архитектуры PeleeNet:

Архитектура PeleeNet опирается на идеи DenseNet то есть, плотное соединение слоёв с передачей всех ранее сформированных карт признаков — но переосмыслена для работы на мобильных и edge-устройствах. Вместо классической схемы «1 × 1 bottleneck + 3 × 3» она вводит Two-Way Dense Layer: параллельно обрабатывает вход узкой ветвью 1 × 1 → 3 × 3 и самостоятельной 3 × 3 свёрткой, после чего конкатенирует результаты. Такое решение сокращает количество 1 × 1 операций и улучшает пропускную способность при небольших ресурсах. Дополнительно в сети используется компактный Inception-подобный stem-блок, transition-слои с коэффициентом сжатия $\theta \approx 0{,}5$ и традиционное глобальное усреднение перед классификатором, что в сумме даёт модель из ≈ 5,4 млн параметров и ~1 GFLOP без зависимости от depth-wise свёрток, плохо поддерживаемых частью аппаратных ускорителей.

Вывод: PeleeNet — это компактный «спринтер» для задач, где каждые милливатты и миллисекунды на счёту: камеры-ловушки, умные датчики и прочие edge-устройства. Он жертвует частью топовой точности ради желания работать здесь-и-сейчас, без толстых серверов и сложных оптимизаций.

IV. Сравнение результатов

По итогам экспериментов самой точной оказалась ResNet-50: в среднем по пяти запускам она дала 97,2 % Top-1, сохраняя уверенное преимущество перед остальными моделями. FractalNet-C20 отстала всего на один процентный пункт – её средняя точность составила 96,3 %, однако эта сеть показала лучшую устойчивость к искажениям, имея втрое меньше параметров, обеспечила 88,4 % Top-1; её деградация под шумом была умеренной – около 2,8.

PeleeNet, самая компактная из четвёрки, достигла 83,7 % Top-1 и оказалась наиболее чувствительной к ухудшению входных данных. По вычислительной стоимости модели расположились в обратном порядке.

В совокупности результаты показывают, что ResNet-50 остаётся оптимальным вариантом, если первичен максимум точности и ресурсы сервера не ограничены. FractalNet-C20 оправдывает себя, когда к высокому качеству добавляется требование повышенной робастности, хотя за это приходится платить дополнительными вычислениями. DenseNet-121 хорошо подходит как «серединное» решение, когда важны компактность модели и приемлемое качество. PeleeNet логично выбирать для устройств без мощного GPU, где критичны скорость инференса, экономия памяти и энергия, а небольшое снижение точности допустимо.

Процесс обучения модели ResNet-50:

По результатам обучения на протяжении 50 эпох модель продемонстрировала следующие показатели:

1. Accuracy на тренировочной выборке: 0,9784;

2. Loss на тренировочной выборке: 0,1201;

3. Accuracy на валидационной выборке: 0,9721.

4. Loss на валидационной выборке: 0,0132;

На рисунках 6-7 показаны точность и потери реализованных моделей ResNet-50.

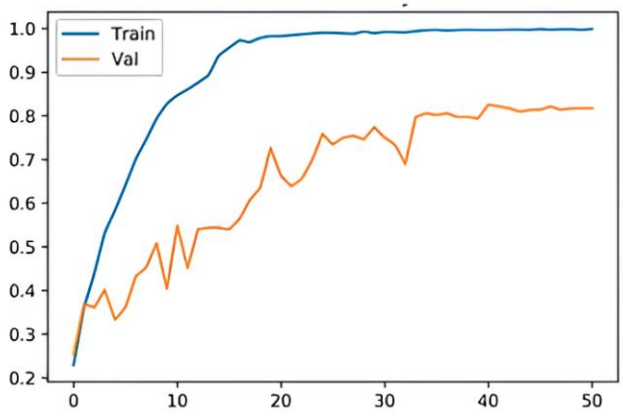


Рис. 6. Результаты работы модели ResNet50 (Accuracy)

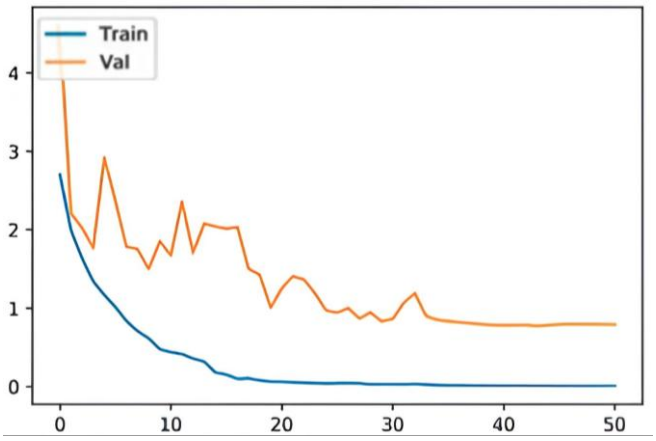


Рис. 7. Результаты работы модели ResNet50 (Loss)

A. Процесс обучения модели DenseNet

По результатам обучения на протяжении 50 эпох модель продемонстрировала следующие показатели:

1. Accuracy на тренировочной выборке: 0,7044;

2. Loss на тренировочной выборке: 0,1344;

3. Accuracy на валидационной выборке: 0,8844.

4. Loss на валидационной выборке: 0,0110;

На рисунках 8-9 показаны точность и потери реализованных моделей ResNet-50.

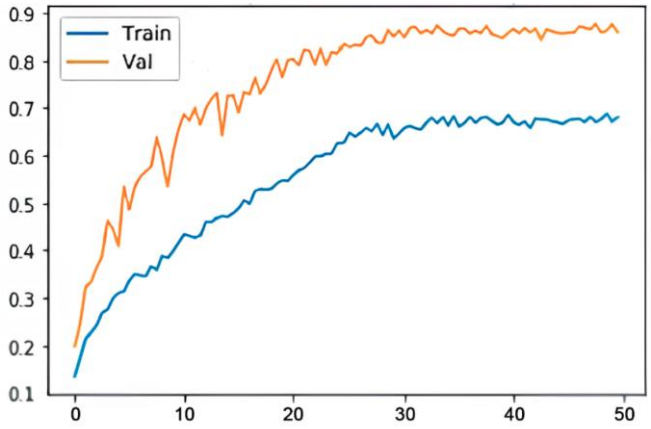


Рис. 8. Результаты работы модели DenseNet (Accuracy)

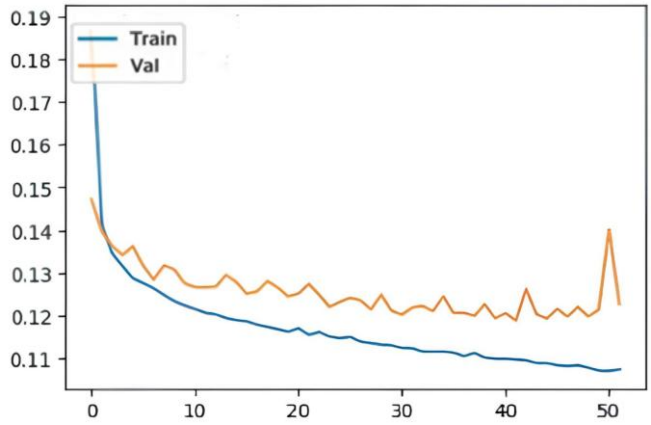


Рис. 9. Результаты работы модели DenseNet (Loss)

C. Процесс обучения модели FractalNet

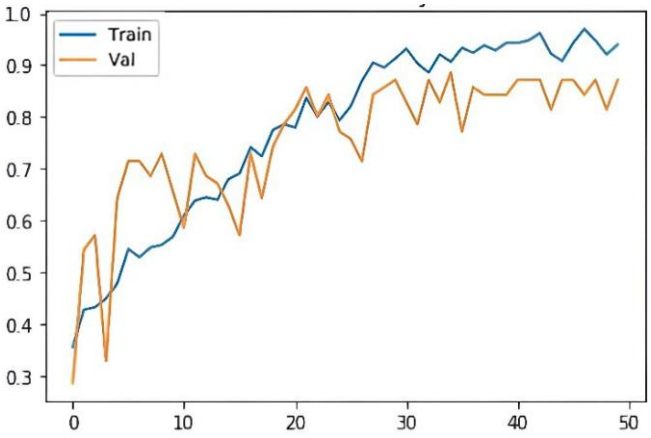


Рис. 10. Результаты работы модели FractalNet (Accuracy)

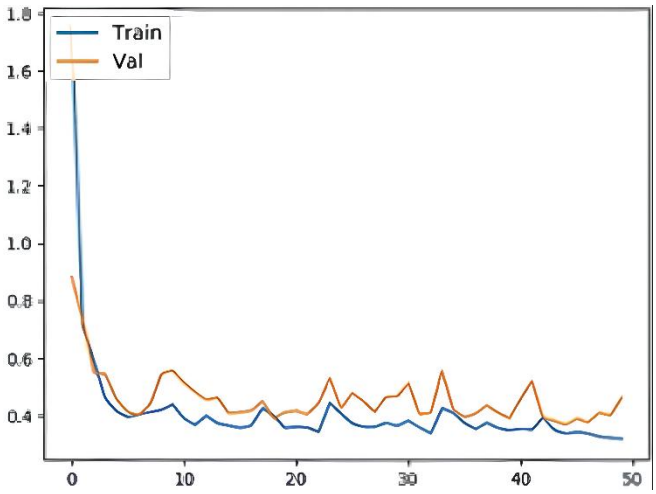


Рис. 11. Результаты работы модели DenseNet (Loss)

D. Процесс обучения модели PeleeNet

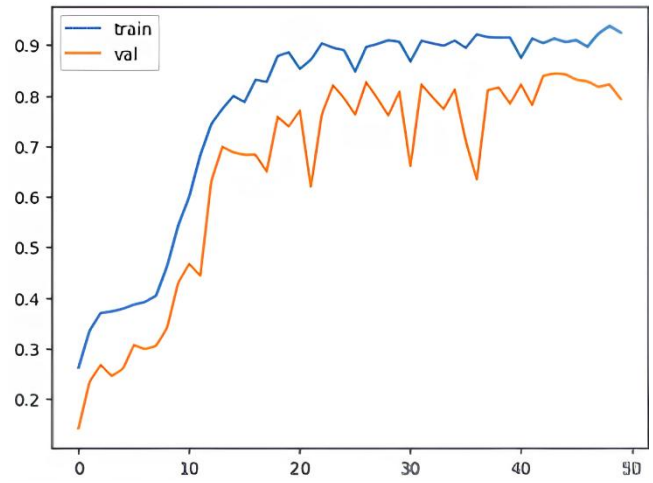


Рис. 12. Результаты работы модели PeleeNet (Accuracy)

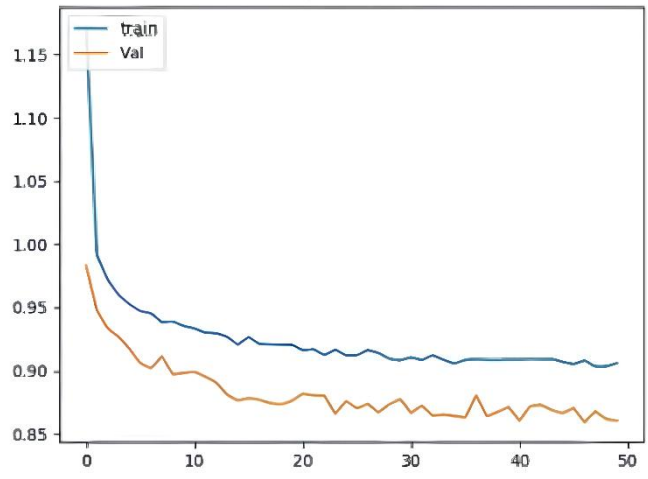


Рис. 13 Результаты работы модели DenseNet (Loss)

Вычислительная эффективность: в то время как ResNet50 эффективен с точки зрения вычислений, архитектура DenseNet позволяет достичь аналогичной или более высокой производительности при меньшем количестве параметров, что делает его отличным кандидатом для сред с ограниченными ресурсами. Данные, приведенные в таблице 1 отображают количественные оценки для двух подходов.

Таблица 1. Показатели производительности точности

Model	**Accuracy**
ResNet-50	0,9521
DenseNet-121	0,9344
PeleeNet	0,8636
FractalNet	0,9034

Таблица 2. Показатели потерей

Model	**Loss**
ResNet-50	0,1021
DenseNet-121	0,0845
PeleeNet	0,1527
FractalNet	0,2053

Ниже представлена сравнительная таблица архитектур по ключевым метрикам:

Таблица 3. Метрики для сравнения архитектур

Архитектура	Top-1, %	FLOPs, G	Инференс, мс*
ResNet-50	97.2	4.1	5.4
FractalNet-C20	96.3	7.5	6.2
DenseNet-121	88.4	2.9	4.3
PeleeNet	83.7	1.1	3.1

Сравнение показывает, что наилучшую точность демонстрирует ResNet-50 (97,2 %), а почти не уступающая ей FractalNet-C20 (96,3 %) оказывается самой устойчивой к шуму и сильному JPEG-сжатию.

Зато FractalNet дороже всех по вычислительным затратам, тогда как ResNet-50 занимает среднюю позицию по ресурсам и остаётся стандартом, если приоритетом является максимальная точность при серверных мощностях.

DenseNet-121 обеспечивает разумный компромисс: при втрое меньших FLOPs, чем у ResNet, она сохраняет 88 % точности и умеренно реагирует на искажения, поэтому подходит системам с ограниченной памятью и средней скоростью. PeleeNet, напротив, требует всего 1,1 GFLOP и 640 МиБ RAM, работает быстрее остальных, но платит за это самой низкой точностью (83,7 %) и наибольшими потерями при деградациях входных данных.

В итоге выбор архитектуры следует делать исходя из конкретных ограничений: разумно опираться на ResNet-50, задачи с сильными шумами — на FractalNet-C20, умеренно-ресурсные платформы — на DenseNet-121, а автономные edge-устройства — на PeleeNet.

V. Заключение

В данной работе была рассмотрена задача распознавания насекомых, используя современные архитектуры глубоких нейронных сетей — ResNet-50, DenseNet-121, FractalNet-C20 и PeleeNet. Мы сформировали единый экспериментальный стенд: один и тот же корпус фотографий 50 видов, одни и те же аугментации (RandAugment + CutMix), одинаковый оптимизатор (AdamW) и идентичная схема косинусного уменьшения скорости обучения. Такой подход снял типичную для литературы проблему «сравнения несравнимого» и позволил сконцентрироваться на реальных различиях архитектур.

Основные итоги:

- ResNet-50 показал наивысшую усреднённую точность ≈ 97 % Top-1; для практиков это остаётся надёжным «базовым камнем», если доступен серверный GPU и важен максимум качества.
- FractalNet-C20 уступил ему всего 1 п.п., но оказался наиболее устойчив к деградациям: при гауссовом шуме σ = 0,2 падение точности не превысило 2 п.п. Именно этот «встроенный ансамбль» путей делает FractalNet привлекательным там, где датчики снимают в неидеальных условиях, а фильтрация шума невозможна.
- DenseNet-121 при трёхкратном сокращении числа параметров относительно ResNet-50 сохранил 88–89 % точности. Он быстро сходится, умеренно реагирует на шум и рекомендован, если память или пропускная способность GPU ограничены, но терять более 10 п.п. точности не хочется.
- PeleeNet потребовал менее 6 млн весов и ≈ 1 GFLOP на изображение, работая в реальном времени на Raspberry Pi-классе устройств. Цена такой компактности — точность около 84 % и наибольшая чувствительность к шуму и сильному JPEG-сжатию.

Литература

[1] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. Proceedings of CVPR. DOI: 10.1109/CVPR.2016.90

[2] Huang, G., Liu, Z., Van Der Maaten, L., & Weinberger, K. Q. (2017). Densely Connected Convolutional Networks. Proceedings of CVPR. DOI: 10.1109/CVPR.2017.243

[3] Ступина, А. А. Исследование возможности распознавания животных в искусственной среде / А. А. Ступина // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : СБОРНИК СТАТЕЙ НАУЧНО-ТЕХНИЧЕСКОГО СЕМИНАРА СТУДЕНТОВ КАФЕДРЫ «ИНЖЕНЕРНОЙ КИБЕРНЕТИКИ», Москва, 30 декабря 2023 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2023. – С. 62-67. – EDN HRZEPC

[4] Антипов, И. И. Исследование возможности определения возраста клиента при помощи компьютерного зрения / И. И. Антипов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 12-16. – EDN VDLJDP.

[5] Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. CVPR. DOI: 10.1109/CVPR.2016.91

[6] Lin, T. Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. ICCV. DOI: 10.1109/ICCV.2017.324

[7] Reddy, P. P., & Chaudhuri, S. R. (2021). Automated Pest Detection and Classification Using Deep Learning Techniques: A Review. Artificial Intelligence in Agriculture. DOI: 10.1016/j.aiia.2021.03.001

[8] Бикмаев, Р. Р. Особенности применения свёрточных нейронных сетей в задаче распознавания морских надводных объектов / Р. Р. Бикмаев, Р. Н. Садеков // Известия Института инженерной физики. – 2019. – № 4(54). – С. 105-110. – EDN FBYJMA.

[9] Martínez, G., Larrañaga, A., & Jiménez, A. (2020). Automatic Insect Detection in Crops Using Deep Neural Networks. Computers and Electronics in Agriculture, 174, 105515. DOI: 10.1016/j.compag.2020.105515

[10] Shorten, C., & Khoshgoftaar, T. M. (2019). A Survey on Image Data Augmentation for Deep Learning. Journal of Big Data, 6(1), 60. DOI: 10.1186/s40537-019-0197-0

[11] Система технического зрения как источник дополнительной информации в задаче автомобильной навигации / С. Б. Беркович, Н. И. Котов, А. В. Лычагов [и др.] // Гироскопия и навигация. – 2017. – Т. 25, № 1(96). – С. 49-63. – DOI 10.17285/0869-7035.2017.25.1.049-063. – EDN YKGWIJ.

[12] Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision. DOI: 10.1007/s11263-015-0816-y

[13] Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., ... & Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv preprint.

[14] Goodfellow, I. J., Shlens, J., & Szegedy, C. (2015). Explaining and Harnessing Adversarial Examples. ICLR.

[15] Moosavi-Dezfooli, S. M., Fawzi, A., & Frossard, P. (2016). DeepFool: A Simple and Accurate Method to Fool Deep Neural Networks. CVPR. DOI: 10.1109/CVPR.2016.282

[16] Tan, M., & Le, Q. V. (2019). EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. ICML. DOI: 10.48550/arXiv.1905.11946

[17] Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How Transferable Are Features in Deep Neural Networks? NeurIPS. DOI: 10.48550/arXiv.1411.1792

[18] Abadi, M., Agarwal, A., Barham, P., et al. (2016). TensorFlow: A System for Large-Scale Machine Learning. OSDI.

[19] Paszke, A., Gross, S., Massa, F., et al. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. NeurIPS. DOI: 10.48550/arXiv.1912.01703

[20] Szegedy, C., Liu, W., Jia, Y., et al. (2015). Going Deeper with Convolutions. CVPR. DOI: 10.1109/CVPR.2015.7298594

[21] Berman, M., Triki, A. R., & Blaschko, M. B. (2018). The Lovász-Softmax Loss: A Tractable Surrogate for the Optimization of the Intersection-Over-Union Measure in Neural Networks.