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, 2026

УДК 0004.8
ББК 32.813.5

Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях, 2026: Сборник статей научно-технического семинара. Вып. 5 / Под ред. А.Р. Ефимова - М.: НИТУ «МИСИС», 2026. - 153 с.: табл., ил., цв.ил.

Не осталось научных сфер, где искусственный интеллект не применяется. Даже философы используют ИИ для генерации идей. Но наибольшее влияние он оказывает в тех областях, где обрабатывается большие объемы данных. Сейчас такие объемы накапливаются не только в физике высоких энергий, но и в биологии, материаловедении и в других дисциплинах.

Ефимов А.Р.
Ведомости «Искусственный интеллект открывает ученым новые горизонты»

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 26 декабря 2025 г.
Редакционная коллегия: Ефимов А.Р., Бакулев К.С., Садеков Р.Н., Мишуров С.С.
Редактор: 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 28.23.15, 28.23.37
ASJC 1702, 1707
OECD 1.02.EP

Определение архитектурного стиля здания по фотографии

А. Р. Алькина
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2508848@edu.misis.ru

***Аннотация* — В статье рассматривается проблема автоматического определения архитектурного стиля зданий — задача, имеющая существенную значимость для исторических исследований, реставрации, градостроительства, туризма и формирования интеллектуальных каталогов объектов культурного наследия. Данный подход позволяет преодолеть ограничения традиционной ручной классификации, которая требует высокой квалификации экспертов и существенных временных затрат. Для решения поставленной задачи разработан метод, основанный на анализе фотографий зданий с применением сверточной нейронной сети MobileNet. В результате исследования была обучена нейронная сеть, а также произведена оценка результатов обучения с помощью метода GradCAM.**

Ключевые слова — архитектурные стили, нейронные сети, классификация, компьютерное зрение.

I. Введение

Задача автоматического определения архитектурного стиля зданий имеет существенную значимость для ряда прикладных областей: исторические исследования, реставрация, градостроительство.

В *исторических исследованиях* автоматизированное распознавание стилей позволяет быстро анализировать большие массивы данных и выявлять закономерности эволюции архитектурных форм в разных регионах.

В *сфере реставрации* технология дает возможность оперативно определять исходный стиль здания при подготовке проектной документации, а также подбирать аутентичные материалы и техники восстановления на основе типологических характеристик стиля.

Для *градостроительства* автоматическое распознавание важно при оценке стилистической гармонии новой застройки с существующим архитектурным контекстом и в разработке регламентов сохранения исторического облика районов.

В настоящее время задача решается привлечением экспертов, что, во-первых, требует ресурсов, а во-вторых, занимает большое количество времени, поэтому автоматизация определения архитектурного стиля является актуальной задачей.

На данный момент задача не имеет возможных автоматизированных решений, кроме использования глубокого обучения, так как определение стиля здания — комплексный процесс, при котором нужно учитывать множество факторов.

Глубокое обучение с использованием сверточных нейронных сетей позволяет выделять признаки с фотографии и по совокупности этих признаков дает возможность определить архитектурный стиль здания. Методы глубокого обучения показали высокую производительность и способность к обобщению во многих областях и типах задач, таких как классификация и обнаружение [1].

MobileNet является семейством сверточных нейронных сетей, разработанных Google с основной целью достижения высокой производительности при ограниченных технических ресурсах. Сверточные нейронные сети широко применяются для решения задач классификации изображений [2].

Таким образом, разработка инструмента автоматического определения архитектурных стилей на основе MobileNet способна повысить эффективность работы специалистов в перечисленных областях за счет сокращения рутинных операций и расширения аналитических возможностей.

II. Обзор предметной области

Архитектурный стиль здания — это совокупность характерных черт и признаков, присущих сооружениям определенной эпохи, места и культуры, отражающая технологии, социальные ценности и эстетические вкусы времени, с такими чертами, как форма, декор, материалы и конструкции. С течением времени стили сменяют друг друга.

Сейчас выделяют несколько основных архитектурных стилей зданий:

- Архитектура Древнего мира: от первобытного общества до V века.
- Раннехристианская архитектура. V—X века.
- Исламская архитектура. V—XXI века.
- Романская архитектура. XI—XII века.
- Нормандская архитектура. XI—XIII века.
- Готика. XIII—XV века.
- Возрождение. Начало XV — начало XVI века.

- Барокко. Середина XVI — середина XVIII века.
- Рококо. Начало XVIII — вторая половина XVIII века.
- Классицизм. Середина XVIII—XIX век.
- Историзм. 1830-е — 1890-е годы.
- Модерн. 1890-е — 1910-е годы.
- Модернизм. Начало 1900-х годов — 1980-е годы.
- Конструктивизм. 1920-е годы — начало 1930-х годов.
- Постмодернизм. С середины XX века.
- Хай-тек. С конца 1970-х годов.
- Деконструктивизм. С конца 1980-х годов.
- Параметризм с 2008 года.

Определение стиля можно производить по нескольким ключевым признакам: форма окон, тип кровли, отделочные материалы, пропорции фасада.

В качестве стилей для классификации (классификация данных с этой точки зрения заключается в нахождении математических границ между классами в пространстве признаков [3, 4]) были выбраны: *классицизм*, *готика*, *барокко* и *модернизм*. Данный выбор обусловлен хорошей документированностью и наличием множества источников изображений, что облегчает сбор разнообразного и репрезентативного датасета. Культурная значимость этих стилей также делает исследование более применимым и интересным для широкой аудитории. Также некоторые стили являются переходными и фактически просто комбинируют черты основных стилей архитектуры.

III. Наборы данных

Для обучения и тестирования модели MobileNet был собран датасет, состоящий из различных фотографий зданий. На каждый архитектурный стиль из выделенных для классификации было собрано около пятисот фотографий в разное время дня и в разные погодные условия.

A. Готический стиль

Для готического стиля характерно изящество, устремленность ввысь, богатое декоративное убранство. Основными элементами стиля принято считать [5]:

- каркасную систему — опорные колонны были заменены на каркасы из арок;
- арочные своды;
- узкие башни с заостренной вершиной — пинакли;
- большие окна, мозаики и порталы.

На рисунке 1 представлен пример здания в готическом стиле.

Рис. 1. Пример здания в готическом стиле

B. Стиль барокко

Готический стиль сменила эпоха Возрождения, а ее в свою очередь сменило появление барокко. Для барокко характерно [6]:

- обилие пышных декоративных украшений;
- подчеркнутая театральность;
- преобладание сложных криволинейных форм;
- ассиметрия;
- использование колонн и полуколонн;
- очень большие порталы, двери и окна.

На рисунке 2 представлен пример здания в стиле барокко.

Рис. 2. Пример здания в стиле барокко

C. Стиль классицизм

В середине 18 века основным архитектурным стилем стал классицизм [7]. Классицизму в архитектуре присущи следующие черты:

- вдохновение античным греческим стилем;
- строгая симметрия;
- простые формы;
- прямые линии;
- ненавязчивый простой декор.

На рисунке 3 представлен пример здания в стиле классицизм.

Рис. 3. Пример здания в стиле классицизм

D. Стиль модернизм

С приходом 20 века одним из наиболее популярных стилей стал модернизм [8]. Он отличается:

- функциональностью и рациональностью;
- геометрической простотой;
- обилием новых материалов;
- малым или вообще отсутствующим декорированием.

На рисунке 4 представлен пример здания в стиле модернизм.

Рис. 4. Пример здания в стиле модернизм

Данные были собраны из различных открытых источников: wiki, unsplash и другие.

В датасете представлены разные снимки, как целых зданий, так и их частей. На рисунках ниже представлены примеры изображений.

Рис. 5. Здание в стиле барокко, представленное не полностью

Рис. 6. Здание в стиле барокко ночью

Рис. 7. Здание в стиле барокко в пасмурную погоду

Рис. 8. Здание в стиле классицизм вечером

Разметка производилась с помощью инструмента CVAT.ai. Размеченный датасет был опубликован на huggingface [9].

IV. Нейросетевая архитектура

A. MobileNet

MobileNet является семейством сверточных нейронных сетей (CNN), разработанных Google с основной целью достижения высокой производительности при ограниченных вычислительных ресурсах.

Ключевой инновацией MobileNet являются *глубинные разделяемые свертки* (Depthwise Separable Convolutions) [10]. Этот подход декомпозирует стандартную свертку в два отдельных, более эффективных шага.

Глубинная свертка применяет один фильтр 3x3 к каждому отдельному входному каналу. Это позволяет

уловить пространственные корреляции без смешивания информации между каналами [10].

Точечная свертка представляет собой свертку 1x1, которая затем комбинирует выходы глубинной свертки. Эта операция отвечает за смешивание информации между каналами, эффективно создавая новые признаковые карты [10].

MobileNet способна эффективно извлекать иерархические визуальные признаки. В контексте архитектуры это включает распознавание общих геометрических паттернов (формы окон, арок, крыш), текстур материалов, наличия и характера декоративных элементов (колонны, пилястры, лепнина) и общих композиционных решений, которые являются определяющими для различных стилей. Также для MobileNet есть предобученные веса ImageNet [11], что позволяет эффективно и быстро дообучить ее на наборе изображений зданий. Обученные на таких данных, CNN способны обнаруживать иерархические и абстрактные признаки, что делает их мощными инструментами для решения разнообразных задач в области компьютерного зрения [2].

На рисунке 9 представлена схема глубинной свертки.

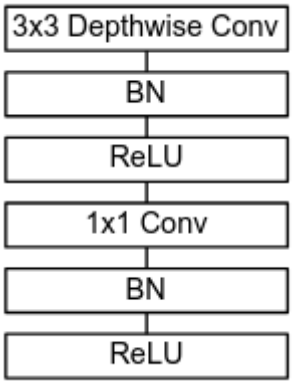


Рис. 9. Схема глубинной свертки

На рисунке 10 представлена схема нейронной сети MobileNet.

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Рис. 10. Схема MobileNet

V. Обучение

Данные для обучения модели были разделены на *train* и *validation* части в соотношении 80 на 20. К ним были применены различные преобразования (*аугментация — данная техника машинного обучения позволяет сгенерировать новые данные за счет имеющихся* [12]): поворот, отражение и т.п.

Модель MobileNet загружалась с предобученными на наборе ImageNet весами, но с отключенными полносвязными слоями модели для дальнейшего обучения на классах архитектурных стилей. Таким образом, используется *transfer learning* — подход, при котором берется частично предобученная модель, которая затем дообучается под конкретную задачу. Далее модель было достроена с помощью добавления сжатия выходных данных предобученной сети, добавления Dropout для защиты от переобучения и добавления полносвязного слоя-классификатора.

Для обучения использовались:

- *оптимизатор Adam*;
- функция потерь *категориальная кроссэнтропия*;
- метрика точность (*accuracy*).

Модель обучалась с ранней остановкой и автоматическим снижением шага обучения, если возникает переобучение.

Графики обучения показаны на рисунках 11-12.

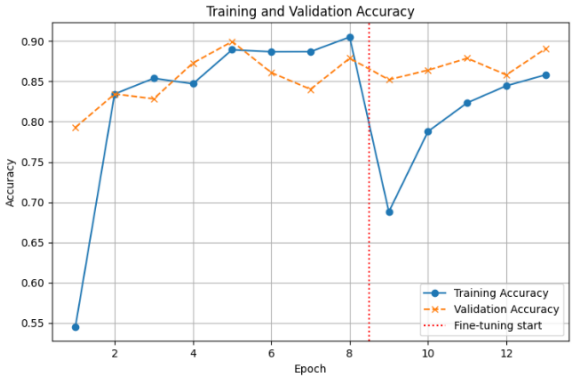


Рис 11. График точности на валидационной и обучающей выборке

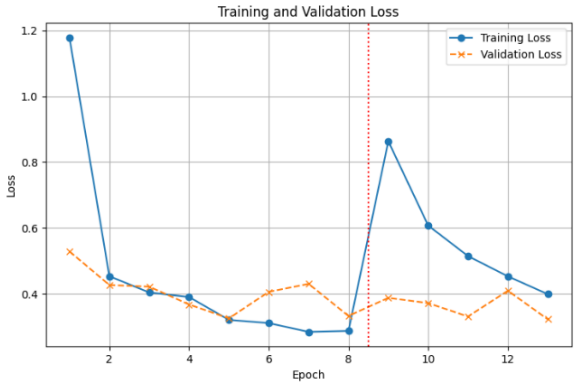


Рис. 12. График функции потерь на валидационной и обучащей выборке

В результате обучения модели MobileNet для четырех выделенных архитектурных стилей получилось добиться точности около 90% на валидационной выборке.

VI. Анализ результатов

Для анализа результатов работы модели была реализована возможность получить *тепловую карту* для классифицируемого изображения, чтобы определить, с помощью каких признаков модель относит изображение здания к тому или иному архитектурному стилю.

Для этого был использован Grad-CAM (Gradient-weighted Class Activation Mapping) — метод визуализации того, на какие области изображения обращает внимание сверточная нейронная сеть при принятии решения [13]. Grad-CAM показывает тепловую карту, где:

- красные/жёлтые зоны — области, сильно влияющие на предсказание;
- синие/зелёные зоны — малозначимые области.

Grad-CAM позволяет получить тепловую карту с помощью нескольких шагов [13].

1. На прямом проходе для изображения обученной нейронной сетью вычисляется предсказание.
2. На обратном проходе вычисляются градиенты оценки класса относительно карт признаков выбранного сверточного слоя.
3. Далее градиенты усредняются по пространственным координатам — получаются веса для каждой карты признаков.
4. Затем карты признаков умножаются на веса и складываются — получается грубая тепловая карта, так как сверточные слои обычно меньше исходного изображения.
5. На последнем этапе карта интерполируется до размера исходного изображения и накладывается на него.

На рисунках 13-16 приведены примеры работы нейронной сети и полученных тепловых карт.

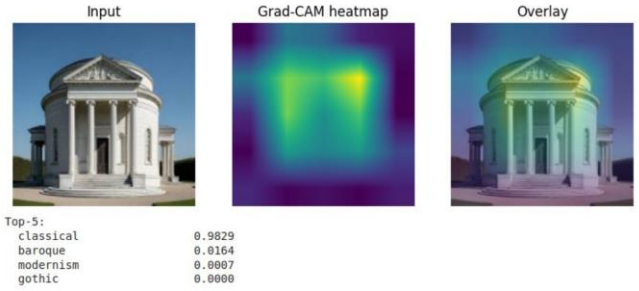


Рис. 13. Тепловая карта для здания в стиле классицизм

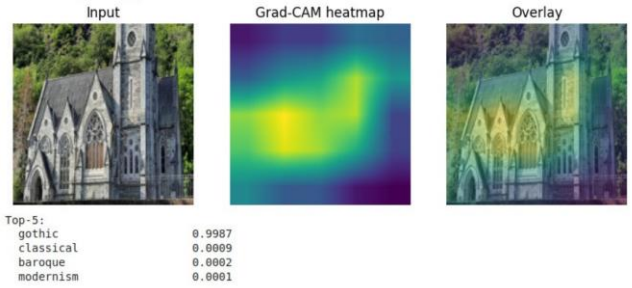


Рис. 14. Тепловая карта для здания в готическом стиле

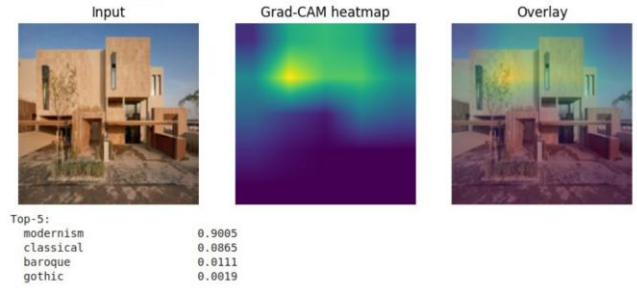


Рис. 15. Тепловая карта для здания в стиле модернизм

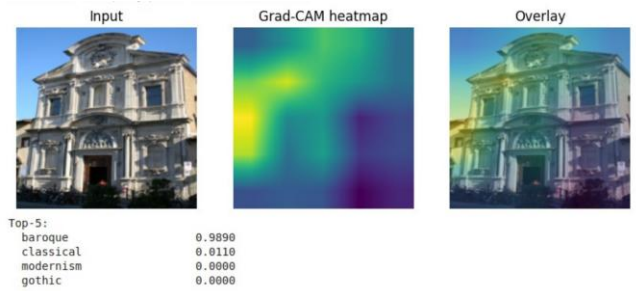


Рис. 16. Тепловая карта здания в стиле барокко

VII. Заключение

В результате обучения модели MobileNet для четырех выделенных архитектурных стилей получилось

добиться точности 87% на валидационной выборке. Полученные результаты были проанализированы с помощью метода получения тепловых карт Grad-CAM. Исходя из составленных тепловых карт, можно сделать вывод, что модель действительно «обращает внимание» на значимые для определения зданий области: колонны, окна, своды крыш и другие.

В качестве дальнейшего развития исследования можно предложить расширение списка классифицируемых стилей, а также увеличение датасета и исследование других архитектур сверточных нейронных сетей в качестве альтернативы MobileNet.

Литература

[1] 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.

[2] Леонов, И. Ю. Классификация транспортных средств компьютерным зрением / И. Ю. Леонов // 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. 46-52. – EDN JSRESQ.

[3] Заречнев, Д. В. Классификация спектров растений методами машинного обучения / Д. В. Заречнев, И. И. Курочкин // Облачные и распределенные вычислительные системы в электронном управлении. ОРВС - 2023 : сборник трудов 4-й международной научно-технической конференции, Переславль-Залесский, 28 ноября – 01 2023 года. – Курск: ЗАО «Университетская книга», 2024. – С. 41-43. – EDN YTKQAV.

[4] Куприянов, В. В. Автоматизация распознавания нештатных ситуаций в угольных шахтах на основе нейронной сети с изменяемыми топологией и весовыми коэффициентами / В. В. Куприянов // Нейрокомпьютеры и их применение : XVIII Всероссийская научная конференция. Тезисы докладов, Москва, 17 марта 2020 года. – Москва: Московский государственный психолого-педагогический университет, 2020. – С. 59-60. – EDN CEVVHF.

[5] Кулагина, Т. О. Готический стиль в архитектуре / Т. О. Кулагина // Вестник ННГАСУ. — Нижний Новгород, 2023. — № 3. — С. 112–120.

[6] *Вёльфлин, Г.* Ренессанс и барокко / Г. Вёльфлин ; переводчик Е. Г. Лундберг ; под редакцией А. Л. Волынского. — Москва : Издательство Юрайт, 2025. — 169 с. — (Антология мысли). — ISBN 978-5-534-12514-6.

[7] Отражение классицизма в мировой архитектуре как искусства, наиболее полно выражающего идеи своего времени : 17.00.04 / Камзина Айгуль Ертисовна - Барнаул, 2016. - 209 с

[8] Ефимов Даниил Дмитриевич, Фахрутдинова Инесса Алековна Истоки и направления советского модернизма // Известия КазГАСУ. 2018. №1 (43). URL: https://cyberleninka.ru/article/n/istoki-i-napravleniya-sovetskogo-modernizma (дата обращения: 27.12.2025).

[9] Набор данных architecture / Hugging Face. — URL: https://huggingface.co/datasets/nastysh-a/architecture (дата обращения: 27.12.2025).

[10] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto and H. Adam, «MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications», arXiv:1704.04861 [cs.CV], 2017.

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

[12] Елисеев, А. Н. Решение задачи классификации сельскохозяйственных культур с использованием глубоких нейронных сетей / А. Н. Елисеев, И. И. Курочкин // Облачные и распределенные вычислительные системы в электронном управлении. ОРВС - 2023 : сборник трудов 4-й международной научно-технической конференции, Переславль-Залесский, 28 ноября – 01 2023 года. – Курск: ЗАО «Университетская книга», 2024. – С. 35-40. – EDN BNXSLL.

[13] Selvaraju, R. R. Grad-CAM: Why did you say that? Visual explanations from deep networks via gradient-based localization / R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, D. Batra // Computing Research Repository (CoRR). — 2016. — Vol. abs/1610.02391.

ГРНТИ 28.23.15, 28.23.37, 28.23.39
ASJC 1702, 1707
OECD 01.02 EP

Сравнительный анализ изменений застройки г. Ижевска на основе сегментации ортофотопланов с применением нейросетевой архитектуры U-Net

А. О. Быкова
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2509312@edu.misis.ru

С. С. Рублева
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2517245@edu.misis.ru

***Аннотация* — В исследовании рассматривается подход к решению задачи сегментации ортофотопланов, основанный на применении нейросетевой архитектуры U-net. В качестве исходных данных использовались фрагменты ортофотопланов г. Ижевска за 2019 и 2024 года. Практически значимым результатом исследования является выявление изменений городской застройки за счет сравнения семантических масок, полученных при сегментации снимков нейросетью. Реализация данного подхода позволит более эффективно проводить контроль выполнения планов государственных программ по благоустройству городов.**

Ключевые слова — глубокое обучение, картография, компьютерное зрение, ортофотоплан, сегментация, семантические маски.

I. Введение

Компьютерное зрение применяется в различных сферах деятельности [1,2], в том числе в области анализа пространственных данных. Актуальность применения данной технологии обуславливается непрерывно растущим спросом на оперативно обновляемые геоданные. Технологии компьютерного зрения и глубокого обучения позволяют получить и обработать большой объем информации с высокой скоростью и точностью, что сокращает сроки реализации геоинформационных проектов. Помимо этого, минимизируется объем ручного труда операторов, и как следствие уменьшается количество ошибок, возникающих из-за человеческого фактора, что повышает эффективность и результативность выполнения работ.

Поддержание актуальных пространственных данных позволяет реализовывать и контролировать выполнение национальных программ в области строительства и благоустройства города. В рамках государственного проекта “Инфраструктура для жизни” регулярно обновляемые пространственные данные позволяют оценивать доступность и уровень развития инфраструктуры, а также планировать ее дальнейшие изменения в соответствии с изменением паттернов поведения населения [3]. Все это становится достижимым благодаря возможности вести точный учет ветхого и аварийного жилья, выявлять незаконные постройки, оценивать плотность и качество застройки в режиме, близком к реальному времени.

В контексте национальной программы “Умный город” регулярно обновляемые пространственные данные формируют точную цифровую копию территории, что является реализацией цифрового двойника городской инфраструктуры.

Подходы, основанные на глубоком обучении, требуют большого количества аннотированных данных. Ручная разметка данных является крайне трудоемким и дорогим процессом. С целью повышения эффективности выполнения данного процесса предлагается использовать машинную разметку данных, основанную на обучении нейронных сетей. Одной из наиболее эффективных архитектур для задач сегментации является U-net, которая позволяет определять точные границы объекта, в научном сообществе признана как одно из эталонных решений State-of-the-Art (SOTA) для задач плотного пиксельного предсказания, особенно в областях медицинской визуализации и анализа дистанционных данных. Архитектура U-net построена по симметричной схеме “энкодер-декодер”, включающая пропускаемые соединения. Именно эта реализация позволяет эффективно комбинировать семантический контекст и пространственные детали, что обеспечивает точное определение контуров объектов на детальных ортофотопланах (ОФП).

Целью данной работы является выявление изменений застройки г. Ижевска за счет решения задачи сегментации ОФП с помощью применения глубокого обучения на основе архитектуры U-net.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались данные предоставленные компанией АО “УСГИК”, которая специализируется на создании геоинформационных продуктов: от аэрофотосъемки и стереофотограмметрической съемки в крупных масштабах до составления и обновления карт средних масштабов по космическим снимкам, от разработки профессионального программного обеспечения до производства стереомониторов [4].

Набор данных представляет собой фрагменты ОФП, полученные в результате аэрофотосъемки с беспилотного летательного аппарата (БПЛА) самолетного типа [5]. Полет производился на высоте 340 м. Размер пикселя на местности, создаваемого по аэрофотоснимкам - 5 см. Масштаб ОФП: 1:500. Точность определения местоположения объекта составляет 25 см.

На снимках представлена застройка города Ижевска (Удмуртская республика, Россия). Пример исходных фрагментов ОФП показан на рисунке 1.

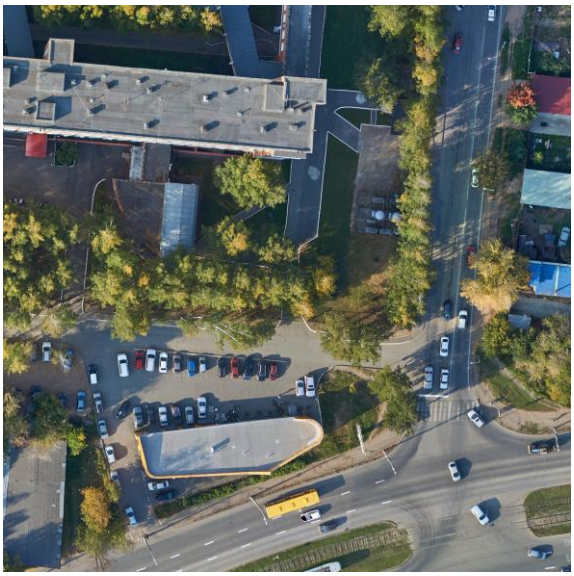


Рис. 1. Пример исходных фрагментов ОФП

Общий набор данных содержит 890 фрагментов ОФП и состоит из двух групп: снимки полученные в результате аэрофотосъемки в 2019 году, и снимки, полученные в результате аэрофотосъемки в 2024 году. Отобраны фрагменты, на которых показана одна и та же территория в одно и то же время года. Благодаря этому становится возможным провести анализ местности на предмет изменений, которые произошли в течение 5 лет.

III. Подготовка данных

Подготовка данных проводилась в два этапа: ручная разметка данных и распознование с помощью нейронных сетей.

Для разметки данных вручную использовалась платформа CVAT.ai [6], которая ориентирована на задачи компьютерного зрения и машинного обучения. С помощью предложенных инструментов разметки выделялась область снимка, которой присваивался один из лейблов: многоэтажный дом (multi-storey building), сельский дом (rural house), гаражи (garages), придомовая территория (house territory), парковочные места (parking lots), дорога (road), деревья (trees), трава (grass), очищенная земля (cleared land), строительная площадка (construction site). Весь снимок был разделен на сегменты, а каждый пиксель изображения принадлежал определенной категории. Пример размеченного фрагмента ОФП представлен на рисунке 2. Таким образом, посредством ручной аннотации были размечены 50 фрагментов ОФП: 25 снимков из выборки за 2019 год и столько же из выборки за 2024 год.

Рис. 2. Пример разметки фрагментов ОФП с помощью инструментов платформы CVAT

Для того, чтобы разметить оставшиеся 840 снимков проводилось обучение нейросетей на основе аннотированных 50 снимков.

Нейросетевая архитектура обучалась для разметки фрагментов ОФП за 2019 год и для разметки фрагментов ОФП за 2024 год. Размеченный датасет опубликован на платформе HuggingFace [7].

IV. Нейросетевая архитектура

Для решения задачи семантической сегментации объектов на аэрофотоснимках использовалась архитектура U-Net [8], которая была реализована на фреймворке PyTorch [9].

Выбор данной архитектуры обусловлен её эффективной симметричной структурой типа «энкодер-декодер» с пропускаемыми соединениями (skip-connections), которые позволяют комбинировать семантические признаки высокого уровня с пространственными деталями низкого уровня, что критически важно для точного восстановления границ объектов при работе с ортофотопланами высокого разрешения.

Модель реализована строго в соответствии с классической U-образной архитектурой (см. рисунок 3). Энкодер состоит из 4 последовательных блоков, каждый из которых включает две свертки 3×3 с дополнением единичным бордюром (padding=1), за которыми следуют операция пакетной нормализации (Batch Normalization) и активация ReLU. После каждого блока применяется операция макс-пулинга с окном 2×2 и шагом 2, что приводит к уменьшению пространственного разрешения в два раза и увеличению количества карт признаков: 64, 128, 256, 512 соответственно.

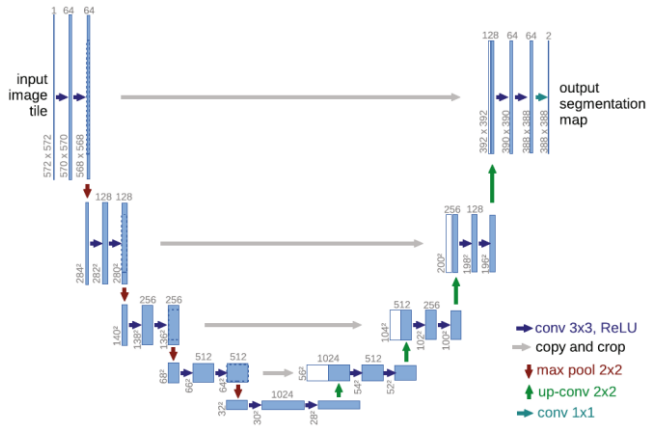


Рис. 3. Строение архитектуры U-net

В области бутылочного горлышка (bottleneck) выполняются два сверточных блока без пулинга, увеличивая размерность признаков до 1024 каналов. Декодер (расширяющий путь) симметричен энкодеру и также состоит из 4 уровней. На каждом уровне сначала применяется транспонированная свертка 2×2 с шагом 2 для увеличения пространственного разрешения, затем результат конкатенируется с соответствующими по размеру картами признаков из энкодера через пропускаемые соединения. После конкатенации выполняются две свертки 3×3 с нормализацией и активацией. Финальный слой представляет собой свертку 1×1, которая проецирует 64-канальную карту признаков на карту логитов размерности C × H × W, где C = 11 (10 целевых классов объектов + фон), H = W = 512.

Для обучения использовалась комбинированная функция потерь L, которая рассчитывается по формуле (1). Данная функция объединяет взвешенную кросс-энтропию и коэффициент Dice Loss, что является стандартной практикой для задач сегментации с несбалансированными классами:

$$L = L_{CE} + L_{Dice} \quad (1)$$

Функция кросс-энтропии L_{CE} вычисляется по формуле (2) с игнорированием пикселей фона, что позволяет модели фокусироваться на релевантных классах объектов:

$$L_{CE} = -\frac{1}{N}\sum_{i=1}^{N} \sum_{c=1}^{C} w_c \cdot y_{i,c} \cdot log(\hat{y}_{i,c}) \quad (2)$$

где N — общее количество пикселей, C = 11 — количество классов, $y_{i,c}$ - индикатор принадлежности пикселя i классу C, $\hat{y}_{i,c}$ - предсказанная вероятность, w_c - вес класса, заданный для компенсации дисбаланса.

Коэффициент Dice Loss L_{Dice}, который рассчитывается по формуле (3), напрямую оптимизирует меру совпадения между предсказанной и эталонной масками, что особенно эффективно для работы с малыми объектами:

$$L_{Dice} = 1 - \frac{2\sum_{c=1}^{C} |P_c \cap G_c| + \epsilon}{\sum_{c=1}^{C} (|P_c| + |G_c|) + \epsilon} \quad (3)$$

В формуле (3) Pc и Gc — это предсказанные и эталонные маски для класса c, а $\epsilon = 10^{-6}$ — коэффициент сглаживания.

В качестве оптимизатора использовался Adam с начальной скоростью обучения 10^{-4}. Для адаптивного снижения скорости обучения применялся планировщик ReduceLROnPlateau, уменьшающий скорость обучения в 2 раза при отсутствии улучшения функции потерь на валидационной выборке в течение 5 эпох.

V. Практическая реализация

Все исходные фрагменты ОФП и соответствующие им маски ручной разметки приведены к единому размеру 512×512 пикселей. Для изображений использовалась билинейная интерполяция, для масок метод ближайшего соседа, что гарантирует сохранение дискретных значений классов. Для увеличения разнообразия обучающей выборки и предотвращения переобучения применен комплексный пайплайн аугментации, включавший случайные горизонтальные и вертикальные отражения (p=0.5), повороты на углы, кратные 90°, а также коррекцию яркости и контраста в пределах ±20%. Интенсивность пикселей изображений нормализовалась по статистике датасета ImageNet: средние значения [0.485, 0.456, 0.406], стандартные отклонения [0.229, 0.224, 0.225].

Эксперимент проводился в среде Google Colab с использованием графического ускорителя NVIDIA Tesla T4 (16 ГБ VRAM). Модель обучалась на выборке из 20 размеченных фрагментов ОФП (20 для обучения, 5 для валидации). Пример семантической маски, размеченной вручную приведен на рисунке 4.

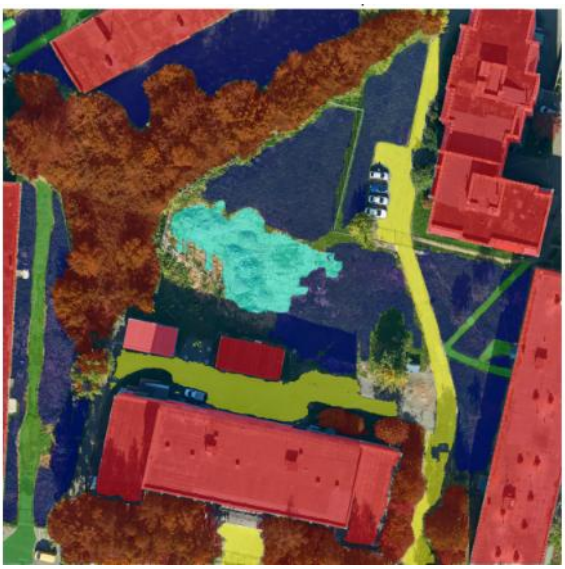


Рис. 4. Пример семантической маски, размеченной вручную

В связи с ограничениями видеопамяти размер батча был установлен равным 2. Для обеспечения стабильности градиентного спроса и имитации большего эффективного размера батча применялась техника накопления градиентов с шагом 2.

Для предотвращения расхождения градиентов использовалось их ограничение с максимальной нормой 1.0.

Обучение проводилось в течение 30 эпох. Наилучшая модель сохранялась на основе мониторинга метрики Intersection over Union (IoU) на валидационной выборке. Максимальное значение Val Loss=2.0844 для

2019 года было достигнуто на 19-й эпохе, а Val Loss=2.1039 для 2024 года на 18-ой эпохе. Полное время обучения составило 7,5 часов на каждый год. Динамика функции потерь на примере выборки за 2019 год представлена на рисунке 5.

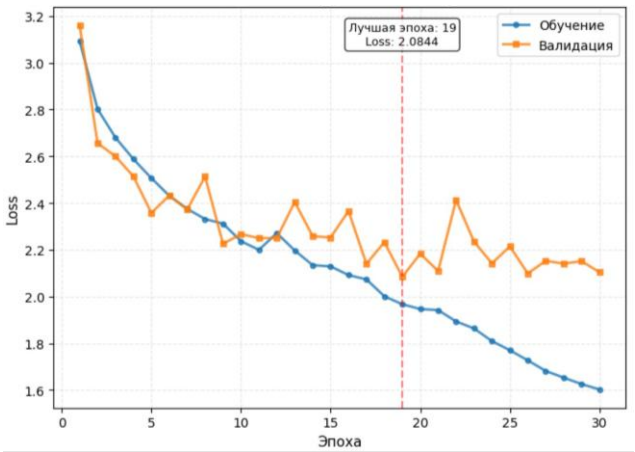


Рис. 5. Динамика функции потерь

Обученная модель была применена для автоматической сегментации полного набора данных, включающего оставшиеся 840 ортофотопланов города Ижевска за 2019 и 2024 годы. Процесс инференса, генерации статистик и визуализаций для всего датасета занял 6 часов [10]. Результатом являются семантические маски в формате PNG, где каждому пикселю сопоставлен идентификатор класса, что обеспечивает основу для последующего точного количественного анализа изменений.

VI. Сравнение результатов сегментации

Для оценки эффективности предложенного подхода было проведено сравнительное исследование результатов семантической сегментации на двух различных наборах данных: фрагменты ОФП 2019 года и фрагменты ОФП 2024 года. Качество работы модели оценивалось с использованием стандартных метрик сегментации, включая Intersection over Union (IoU), Precision, Recall и F1-меру для каждого класса объектов.

Оценка на данных 2019 года. На рисунке 6 представлены результаты семантической сегментации модели U-Net на фрагментах ОФП 2019 года. Визуальный анализ демонстрирует, что модель успешно выделяет основные классы объектов, включая здания, дороги и зеленые насаждения. Однако наблюдаются некоторые ошибки сегментации на границах объектов, а также неполное выделение объектов сложной формы. Среднее значение IoU по всем классам составило 0.343, что указывает на удовлетворительное качество работы модели на данных 2019 года.

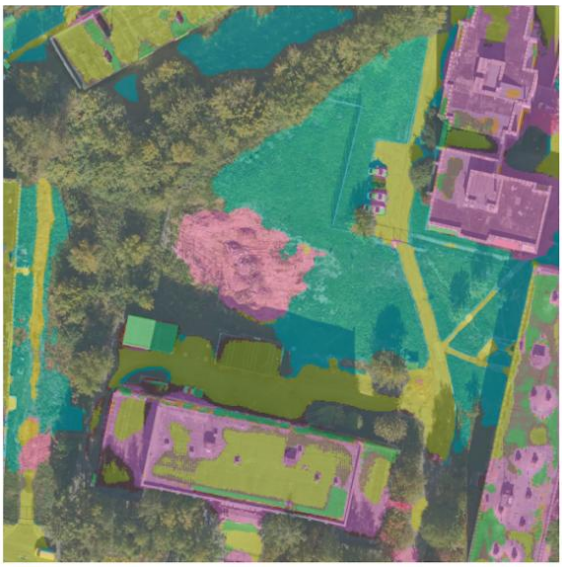


Рис. 6. Семантическая сегментация модели U-Net на данных за 2019 год

Оценка на данных 2024 года. На рисунке 7 показаны результаты сегментации на более современных ортофотопланах 2024 года. Можно отметить значительное улучшение качества сегментации по сравнению с данными 2019 года. Модель демонстрирует более точное выделение границ объектов, лучше справляется с объектами сложной геометрии и более стабильно выделяет малые объекты. Среднее значение IoU равно 0.286, что свидетельствует об удовлетворительной адаптации модели к особенностям современных данных.

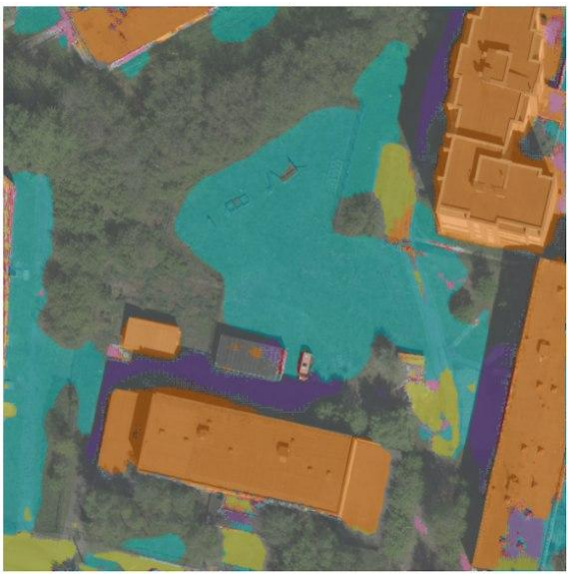


Рис. 7. Семантическая сегментация модели U-Net на данных за 2024 год

Для объективной оценки использовались следующие метрики:

- $Precision = \frac{TP}{TP+FP}$ – доля площади, корректно классифицированной моделью как данный класс, относительно всей площади, которую модель отнесла к этому классу;

- $Recall = \frac{TP}{TP+FN}$ – доля площади эталонного класса, корректно обнаруженная моделью, относительно всей площади этого класса в ручной разметке;
- $F1 = 2\frac{Precision \cdot Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN}$ – оценка баланса между точностью (precision) и полнотой (recall).
- $IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP+FP+FN}$ - коэффициент Жаккара обозначающий меру площади пересечения предсказанной и эталонной масок в отношении к площади их объединения

В приведенных формулах используются следующие обозначения: TP – количество правильных пикселей, которые и модель, и эксперт-разметчик отнесли к нужному классу; FP – количество пикселей, которые модель посчитала ошибочно, отнесла модель к целевому классу; FN – количество пропущенных пикселей целевого класса.

В таблице 1 представлено сравнение метрик для двух наборов данных. Значительное улучшение всех метрик на данных 2024 года объясняется несколькими факторами: более высоким разрешением исходных ортофотопланов, улучшенным качеством разметки, а также адаптацией аугментационных техник к специфике современных данных.

Таблица 1 - Сравнение метрик для наборов данных за 2019 и 2024 года

Лейбл	Год	Recall	Precis.	F1	IoU
многоэтаж. дом	2019	0,55	0,60	0,57	0,38
	2024	0,12	0,18	0,14	0,08
сельский дом	2019	0,58	0,62	0,60	0,45
	2024	0,42	0,48	0,45	0,32
гаражи	2019	0,04	0,10	0,06	0,13
	2024	0,02	0,05	0,03	0,14
придомовая территория	2019	0,46	0,32	0,38	0,23
	2024	0,43	0,53	0,48	0,38
парковоч. места	2019	0,36	0,40	0,38	0,37
	2024	0,37	0,42	0,39	0,40
дорога	2019	0,29	0,33	0,31	0,37
	2024	0,34	0,26	0,29	0,22
деревья	2019	0,65	0,70	0,67	0,61
	2024	0,55	0,49	0,52	0,49
трава	2019	0,41	0,39	0,40	0,38
	2024	0,40	0,34	0,37	0,31
очищенная земля	2019	0,43	0,48	0,45	0,48
	2024	0,41	0,33	0,36	0,33
строит. площадка	2019	0,02	0,03	0,02	0,03
	2024	0,18	0,20	0,19	0,19

VII. Получение результатов изменения застройки

Для выявления изменений были сопоставлены пары семантических масок, полученных для ортофотопланов 2019 и 2024 годов. Различия выявлялись путем прямого сравнения значений пикселей: если метка класса в одной и той же позиции различалась для двух дат, пиксель помечался как измененный. На основе этого формировалась общая маска изменений, из которой затем выделялись связные области полигоны изменений.

Для каждого полигона анализировался его семантический состав в оба периода. Определялся преобладающий класс в пределах полигона как для 2019, так и для 2024 года. Сравнение этих классов для одного полигона формировало итоговую характеристику произошедшего изменения, например «деревья → очищенная земля».

Вывод в консоль результата:

```
Обнаружено 363 областей с изменением классов
Область 1672 (125,831 пикс.):
Было: trees (89.5%) -> Стало:  cleared land (90.3%)
Область 1800 (18,849 пикс.):
Было: trees (79.3%) -> Стало:  parking lots (89.9%)
Область 673 (11,346 пикс.):
Было: trees (90.1%) -> Стало:  cleared land (95.4%)
```

Результат отображения выявленного изменения территории приведен на рисунке 8. На данных снимках видно, что в период с 2019 по 2024 год проведена

подготовка территории для проведения строительных работ.

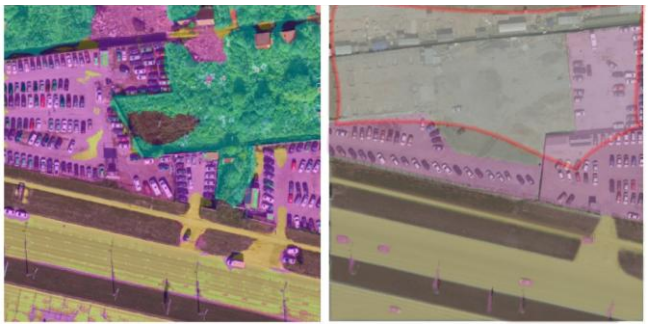


Рис. 8. Выявленная область изменения территории

VIII. Заключение

В работе рассмотрено применение нейросетевой архитектуры U-net для решения задачи сегментации городской застройки.

Для обучения и тестирования модели использовались 890 фрагментов ОФП г. Ижевска, полученные в результате аэрофотосъемки с БПЛА. Датасет размещен в открытом доступе на платформе HuggingFace [7], что обеспечивает воспроизводимость и доступность результатов.

Нейронная сеть U-net рассмотрена с точки зрения реализации архитектуры и особенностей процесса обучения.

Проведен анализ полученных результатов с помощью метрик Precision, Recall, F1 и IoT. Средняя точность модели (IoU) составила 0,34 в 2019 году и 0,29 в 2024 году, что указывает на необходимость доработки модели для современных данных.

Наилучшие результаты достигнуты для классов «деревья» (IoU 0,61 в 2019; 0,49 в 2024) и «очищенная земля» (IoU 0,48 в 2019). Наиболее проблемными оказались классы «строительная площадка» (IoU 0,03 в 2019; 0,19 в 2024) и «гаражи» (IoU 0,13–0,14). Существенное ухудшение в 2024 году отмечено для «многоэтажных домов» (падение IoU с 0,38 до 0,08) и «дорог».

Для повышения точности необходимо расширение обучающей выборки ручной разметкой сложных объектов, а также применение специализированной аугментации и сравнительный анализ с другими архитектурами.

Литература

[1] Андронов, К. В. Сравнение и анализ моделей FCN и DeepLabV3 в задаче семантической сегментации объектов городской улицы / К. В. Андронов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 9-13. – EDN VALUNW.

[2] Каримов, Р. А. Сегментация строений на изображениях с видом сверху / Р. А. Каримов, М. Э. Насибов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 57-62. – EDN XLXNHP.

[3] Citizens' digital infrastructure as a new element of modern society critical infrastructure / L. A. Reingold, E. A. Reingold, A. V. Soloviev, O. S. Grin // Physics and Technology Proceedings (CPT2020) : Conference Proceedings The 8th International Scientific Conference on Computing, 09–13 ноября 2020 года. – Nizhny Novgorod: Автономная некоммерческая организация в области информационных технологий "Научно-исследовательский центр физико-технической информатики", 2020. – P. 128-132. – DOI 10.30987/conferencearticle_5fce2773f10a56.34908891. – EDN ZALSKC.

[4] Урало-Сибирская Гео-Информационная Компания [Электронный ресурс]. URL: https://usgik.ru/ (Дата обращения: 19.12.2025)

[5] Пазычев, Д. Б. Комплекс навигации для беспилотного летательного аппарата / Д. Б. Пазычев, К. С. Бакулев // XXXI Санкт-Петербургская международная конференция по интегрированным навигационным системам : Сборник докладов, Санкт-Петербург, 27–29 мая 2024 года. – Санкт-Петербург: "Концерн "Центральный научно-исследовательский институт "Электроприбор", 2024. – С. 200-207. – EDN PMNQMR.

[6] Leading Image and Data Annotation Platform | CVAT [Электронный ресурс]. URL: https://www.cvat.ai/ (Дата обращения: 19.12.2025)

[7] HuggingFace. Datasets. Izhevsk_OFP [Электронный ресурс] URL:https://huggingface.co/datasets/fyzbykova/Izhevsk_OFP/tree/main (Дата обращения: 27.12.2025)

[8] U-net Documentation [Электронный ресурс] URL: https://u-net.readthedocs.io/_/downloads/en/latest/pdf/(Дата обращения: 25.12.2025)

[9] PyTorch [Электронный ресурс] URL: https://pytorch.org/(Дата обращения: 25.12.2025)

[10] Глущенко, А. И. Об определении предельной скорости обучения нейронной сети для задачи настройки параметров ПИ-регулятора при управлении нагревательными объектами / А. И. Глущенко // Международная конференция по мягким вычислениям и измерениям. – 2017. – Т. 1. – С. 509-512. – EDN ZDFVEB.

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

Разработка интерактивной драм-машины на основе распознавания жестов рук

Б. В. Варецкий
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2512519@edu.misis.ru

К. А. Проскуряков
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2512597@edu.misis.ru

***Аннотация* — в работе решается задача распознавания статических жестов рук для управления виртуальной драм-машиной. Система использует библиотеку MediaPipe для детекции ключевых точек рук и многослойный персептрон для классификации десяти различных жестов. Собран собственный датасет из 1996 примеров. Обученная модель достигла точности 94% на тестовом наборе данных при размере 10 килобайт. Реализовано веб-приложение на базе Next.js с распознаванием жестов в режиме реального времени и задержкой 40-60 миллисекунд.**

Драм-машина представляет собой электронный музыкальный инструмент для синтеза звуков ударных инструментов. Традиционные драм-машины требуют физического оборудования или сложных программных интерфейсов, что создает барьер для начинающих пользователей.

Ключевые слова — компьютерное зрение, распознавание жестов, машинное обучение, mediapipe, tensorflow lite, нейронные сети, веб-приложение

I. Введение

Распознавание жестов рук является активно развивающейся областью компьютерного зрения с применением в человеко-машинном взаимодействии, виртуальной реальности и интерактивных системах [1]. Современные методы глубокого обучения позволяют создавать точные системы распознавания, работающие в реальном времени [2].

Существующие подходы к распознаванию жестов можно разделить на несколько категорий. Методы, основанные на анализе изображений целиком с использованием сверточных нейронных сетей, обеспечивают высокую точность, но требуют значительных вычислительных ресурсов. Методы детекции ключевых точек рук с последующей классификацией позволяют получить компактные модели с высокой скоростью работы. Гибридные подходы комбинируют различные техники для достижения баланса между точностью и производительностью [3].

MediaPipe представляет собой open-source фреймворк от Google, предоставляющий готовые решения для детекции и отслеживания рук. Библиотека использует машинное обучение для определения 21 ключевой точки рук в трехмерном пространстве. Преимуществом MediaPipe является высокая скорость работы и возможность развертывания как на мобильных устройствах, так и в веб-браузерах.

Технология распознавания жестов рук основана на методах компьютерного зрения и глубокого обучения. Современные подходы используют сверточные нейронные сети для извлечения признаков из изображений [4], рекуррентные сети для анализа временных последовательностей [5], а также гибридные архитектуры, сочетающие различные типы слоев [6]. Детекция ключевых точек рук может выполняться с помощью специализированных моделей, таких как OpenPose [7] и MediaPipe [8].

Классификация жестов после извлечения признаков осуществляется различными методами машинного обучения. Простые классификаторы, такие как метод k-ближайших соседей и метод опорных векторов, показывают хорошие результаты на небольших датасетах [9]. Для более сложных задач применяются глубокие нейронные сети различных архитектур [10].

Актуальность данной работы обусловлена растущим интересом к интерактивным музыкальным приложениям и необходимостью создания интуитивных интерфейсов для музыкального творчества. Драм-машины традиционно требуют физического оборудования или сложных программных интерфейсов, что создает барьер для начинающих пользователей. Использование распознавания жестов позволяет создать естественный и доступный способ взаимодействия с виртуальными музыкальными инструментами.

Целью данной работы является разработка системы распознавания жестов рук для интерактивной драм-машины с интеграцией в веб-приложение.

II. Наборы данных

Для обучения модели классификации был собран собственный датасет, содержащий десять классов жестов рук. Сбор данных осуществлялся с помощью специального скрипта на Python, позволяющего сохранять предобработанные координаты ключевых точек при нажатии клавиш от нуля до девяти.

Датасет включает следующие классы жестов:

- один палец вверх;
- два пальца вверх;
- три пальца вверх;
- четыре пальца вверх;

- раскрытая ладонь;
- знак окей;
- кулак;
- жест rock с прижатым большим пальцем;
- жест L;
- открытая ладонь.

Примеры жестов с распознанными классами отображены на рисунке 1.

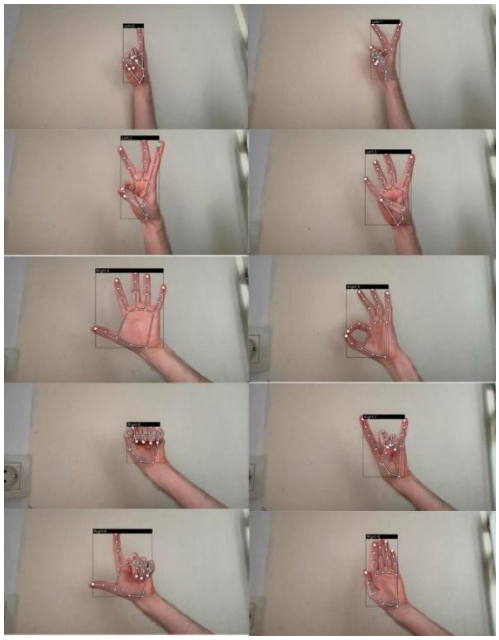


Рис. 1. Жесты и их классы

Для каждого класса было собрано от 132 до 268 примеров. Общий объем датасета составил 1996 образцов. Распределение примеров по классам показано на рисунке 2.

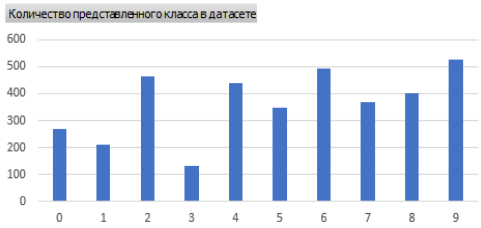


Рис. 2. Распределение примеров по классам датасета

При сборе данных использовалась правая рука в различных положениях относительно камеры для обеспечения инвариантности к небольшим поворотам и смещениям.

Датасет опубликован в открытом доступе на платформе Hugging Face для возможности воспроизведения результатов и дальнейших исследований [11].

III. АРХИТЕКТУРА СИСТЕМЫ

Разработанная система состоит из трех основных компонентов: веб-интерфейс, backend сервер и pipeline обучения модели.

Веб-интерфейс реализован на фреймворке Next.js с использованием TypeScript. Для детекции рук используется MediaPipe Tasks Vision в браузерной WASM версии. Интерфейс отображает видеопоток с веб-камеры, визуализирует обнаруженные ключевые точки рук и показывает распознанный жест с уровнем уверенности модели.

Backend сервер реализован на FastAPI и предоставляет REST API endpoint для классификации жестов. Сервер принимает предобработанные координаты ключевых точек, выполняет inference с помощью TensorFlow Lite модели и возвращает класс жеста с уровнем уверенности.

Pipeline обучения включает скрипт сбора данных на Python и Jupyter notebook для обучения модели. Общая архитектура системы представлена на рисунке 3.

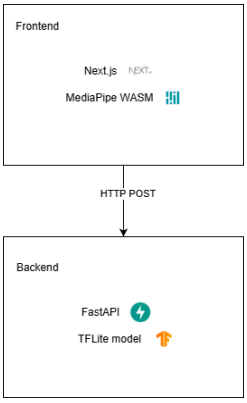


Рис. 3. Общая архитектура системы распознавания жестов

A. *Детекция ключевых точек*

Для детекции рук и извлечения координат ключевых точек используется MediaPipe Hand Landmarker. Модель определяет 21 ключевую точку рук, включая запястье, метакарпальные суставы, проксимальные межфаланговые суставы, дистальные межфаланговые суставы и кончики пальцев.

MediaPipe возвращает нормализованные координаты для каждой точки в диапазоне от нуля до единицы относительно размеров изображения. Помимо двухмерных координат предоставляется третья координата z, представляющая глубину точки относительно запястья. Структура ключевых точек показана на рисунке 3.

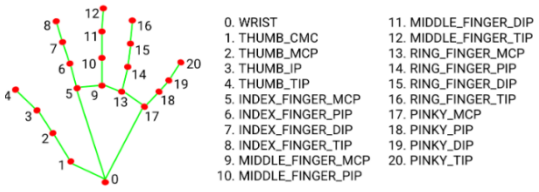


Рис. 3. Ключевые точки руки в MediaPipe

Модель MediaPipe также определяет handedness, то есть принадлежность рук к левой или правой. Эта информация используется для корректной обработки зеркального отображения видеопотока.

B. Предварительная обработка данных

Для приведения данных к формату, пригодному для обучения модели, применяется четырехэтапный preprocessing pipeline.

На первом этапе нормализованные координаты конвертируются в пиксельные координаты изображения размером 640 на 480 пикселей. Координата x умножается на ширину изображения, координата y на высоту изображения, результат округляется до целых значений.

На втором этапе все координаты пересчитываются относительно запястья, являющегося нулевой ключевой точкой. Из каждой координаты вычитается соответствующая координата запястья. Это делает представление инвариантным к положению рук в кадре.

На третьем этапе координаты нормализуются делением на максимальное абсолютное значение среди всех координат. Это приводит значения к диапазону от минус единицы до единицы и делает представление инвариантным к масштабу рук.

На четвертом этапе формируется одномерный вектор признаков путем последовательного расположения координат всех точек. Из 21 точки с двумя координатами получается 42-мерный вектор признаков. Схема preprocessing pipeline показана на рисунке 4.

① Координаты ключевых точек

ID : 0	ID : 1	ID : 2	ID : 3	·····	ID : 17	ID : 18	ID : 19	ID : 20
[551, 465]	[485, 428]	[439, 362]	[408, 307]	·····	[633, 315]	[668, 261]	[687, 225]	[702, 188]

② Относительные координаты от запястья (ID:0)

ID : 0	ID : 1	ID : 2	ID : 3	·····	ID : 17	ID : 18	ID : 19	ID : 20
[0, 0]	[-66, -37]	[-112, -103]	[-143, -158]	·····	[82, -150]	[117, -204]	[136, -240]	[151, -277]

③ Формирование одномерного вектора

ID : 0		ID : 1		ID : 2		ID : 3		·····	ID : 17		ID : 18		ID : 19		ID : 20	
0	0	-66	-37	-112	-103	-143	-158	·····	82	-150	117	-204	136	-240	151	-277

④ Нормализация делением на max(|coords|)

ID : 0		ID : 1		ID : 2		ID : 3		·····	ID : 17		ID : 18		ID : 19		ID : 20	
0	0	-0.24	-0.13	-0.4	-0.37	-0.52	-0.57	·····	0.296	-0.54	0.422	-0.74	0.491	-0.87	0.545	-1

Рис. 4. Предварительная обработка координат ключевых точек

Математически процесс предобработки описывается следующим образом.

На первом этапе нормализованные координаты MediaPipe преобразуются в пиксельные:

$$x_{pixel}(i) = \lfloor x_{norm}(i) \times W \rfloor \quad (1)$$

$$y_{pixel}(i) = \lfloor y_{norm}(i) \times H \rfloor \quad (2)$$

где W = 640 и H = 480 – ширина и высота изображения в пикселях, i =0, 1, …, 20 – индекс ключевой точки, $\lfloor . \rfloor$ - функция округления вниз.

На втором этапе вычисляются относительные координаты:

$$x_{rel}(i) = \left\lfloor x_{pixel}(i) - x_{pixel}(0) \right\rfloor \quad (3)$$

$$y_{rel}(i) = \left\lfloor y_{pixel}(i) - y_{pixel}(0) \right\rfloor \quad (4)$$

где точка 0 соответствует запястью руки.

На третьем этапе применяется нормализация:

$$coords = \begin{bmatrix} x_{rel}(0), y_{rel}(0), x_{rel}(1), y_{rel}(1), \dots \\ , x_{rel}(20), y_{rel}(20) \end{bmatrix} \quad (5)$$

$$max_{val} = \max(|coords|) \quad (6)$$

$$coords_{norm} = \frac{coords}{max_{val}} \quad (7)$$

Финальный вектор признаков имеет размерность 42 и содержит нормализованные относительные координаты всех ключевых точек.

IV. МОДЕЛЬ КЛАССИФИКАЦИИ

Для классификации жестов используется многослойный персептрон со следующей архитектурой. Входной слой содержит 42 нейрона, соответствующих вектору признаков. За ним следует dropout слой с коэффициентом 0.2 для предотвращения переобучения.

Первый скрытый слой содержит 20 нейронов с функцией активации ReLU. После него располагается dropout слой с коэффициентом 0.4. Второй скрытый слой содержит 10 нейронов с функцией активации ReLU.

Выходной слой содержит 10 нейронов, соответствующих классам жестов, с функцией активации softmax для получения вероятностного распределения по классам. Архитектура модели показана на рисунке 5.

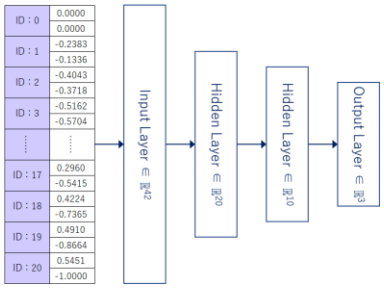


Рис. 5. Архитектура нейронной сети для классификации жестов

Прямое распространение в многослойном персептроне описывается следующими уравнениями.

Для первого скрытого слоя:

$$h_1 = ReLU(W_1 x + b_1) \quad (8)$$

$$h_{1,dropout} = Dropout(h_1, p = 0.2) \quad (9)$$

где x – входной вектор размерности 42, $W_1 \in \mathrm{R}^{20\times 42}$- матрица весов, $b_1 \in \mathrm{R}^{20}$– вектор смещений, $ReLU(z) = \max(0, z)$

Для второго скрытого слоя:

$$h_2 = ReLU(W_2 h_{1,dropout} + b_2) \quad (10)$$

$$h_{2,dropout} = Dropout(h_2, p = 0.4) \quad (11)$$

где $W_2 \in \mathrm{R}^{10\times 20}$– матрица весов, $b_2 \in \mathrm{R}^{10}$– вектор смещений.

Для выходного слоя:

$$z = W_3 h_{2,dropout} + b_3 \quad (12)$$

$$y = softmax(z) \quad (13)$$

где $W_3 \in R^{10\times10}$– матрица весов, $b_3 \in R^{10}$– вектор смещений, softmax активация:

$$softmax(z_i) = \frac{\exp(z_i)}{\sum_{j=1}^{10}\exp(z_i)} \quad (14)$$

Функция потерь sparse categorical crossentropy для обучения:

$$L = -\log(y_{true}) \quad (15)$$

где y_{true} – вероятность правильного класса.

Модель обучалась с использованием оптимизатора Adam и функции потерь sparse categorical crossentropy. Размер батча составлял 128 примеров. Обучение проводилось с ранней остановкой при отсутствии улучшения точности на валидационной выборке в течение 20 эпох.

Данные разделялись на обучающую и тестовую выборки в соотношении 75 к 25 процентам. Обучающая выборка содержала 1497 примеров, тестовая 499 примеров.

TensorFlow Lite представляет собой облегченную версию фреймворка TensorFlow, предназначенную для развертывания моделей машинного обучения на мобильных и встраиваемых устройствах [12]. Основными преимуществами TensorFlow Lite являются малый размер библиотеки, низкая задержка inference и поддержка аппаратного ускорения [13].

V. РЕЗУЛЬТАТ

Обученная модель достигла точности 94.0 процента на тестовой выборке при значении функции потерь 0.2444. Матрица ошибок представлена на рисунке 6.

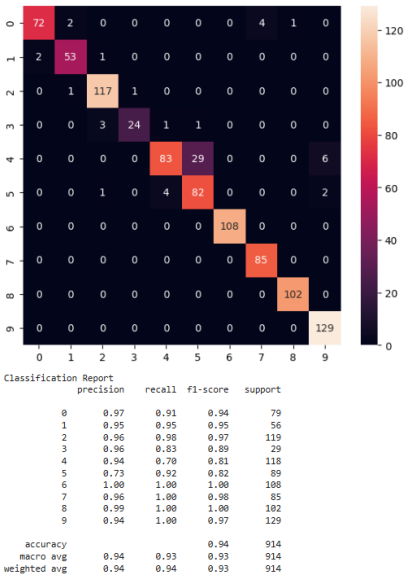


Рис. 6. Матрица ошибок классификации десяти жестов

Детальные метрики по классам показаны в таблице 1. Наиболее точно распознаются жесты кулака, rock и L с F1-мерой от 0.98 до 1.00. Это обусловлено их отличительными признаками и хорошей разделимостью в пространстве признаков.

Наименьшая точность наблюдается для жеста раскрытой ладони класса 4 с F1-мерой 0.81. Это связано с похожестью данного жеста на жест открытой ладони класса 9, что приводит к путанице между этими классами.

ТАБЛИЦА I. Оценка детектирующей части

Класс	Precission	Recall	F1-Score
0	0.97	0.91	0.94
1	0.95	0.95	0.95
2	0.96	0.98	0.97
3	0.96	0.83	0.89
4	0.94	0.70	0.81
5	0.73	0.92	0.82
6	1.00	1.00	1.00
7	0.96	1.00	0.98
8	0.99	1.00	1.00
9	0.94	1.00	0.97

Для оценки качества классификации используются стандартные метрики. Precision для класса cc с вычисляется как:

$$Precission = \frac{TP_c}{TP_c + FP_c} \quad (16)$$

где TP_c – количество истинно положительных примеров класса cc с, FP_c – количество ложноположительных.

Recall для класса с:

$$Recall = \frac{TP_c}{TP_c + FN_c} \quad (17)$$

где FN_c – количество ложноотрицательных примеров.

F1-мера представляет гармоническое среднее precision и recall:

$$F1_c = \frac{2 \times Precission_c \times Recall_c}{Precission_c + Recall_c} \quad (18)$$

Общая точность модели (accuracy) вычисляется как:

$$Accuracy = \frac{\sum_{c=0}^{9} TP_c}{N} \quad (19)$$

где N – общее количество примеров в тестовой выборке.

Производительность системы измерялась в режиме реального времени. Детекция рук MediaPipe занимает от 20 до 30 миллисекунд на кадр. Предобработка координат выполняется менее чем за 1 миллисекунду. Inference модели на backend сервере занимает от 5 до 10 миллисекунд. Сетевой обмен между frontend и backend при локальном развертывании занимает от 10 до 20 миллисекунд.

Общая задержка end-to-end pipeline составляет от 40 до 60 миллисекунд, что обеспечивает комфортное

взаимодействие пользователя с системой в режиме реального времени.

VI. Заключение

В работе представлена полнофункциональная система распознавания жестов рук для интерактивной драм-машины. Разработанное решение включает сбор датасета, обучение модели классификации и интеграцию с веб-приложением.

Собран датасет из 1996 примеров для десяти классов жестов рук.

Обучена модель многослойного персептрона, достигшая точности 94 процента на тестовой выборке. Реализовано веб-приложение с распознаванием жестов в режиме реального времени и общей задержкой от 40 до 60 миллисекунд. Решена проблема зеркального отображения координат landmarks при работе с веб-камерой.

Разработанная система демонстрирует практическую применимость методов компьютерного зрения и машинного обучения для создания интерактивных музыкальных интерфейсов. Подход может быть адаптирован для других приложений человеко-машинного взаимодействия, таких как управление умным домом, виртуальная реальность, бесконтактные интерфейсы.

Литература

[1] Алексеев Л.Е. Распознавание самокатов в реальном времени с помощью YOLO: сравнительный анализ YOLOv10, YOLOv11 // Сборник статей научно-технического семинара кафедры «Инженерной кибернетики» на тему «Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях». М.: НИТУ «МИСИС», 2025. С. 4-7.

[2] Бахвалов И.Б., Кузьменко А.А. Нейросетевые методы для распознавания дефектов в дорожном покрытии // Сборник статей научно-технического семинара кафедры «Инженерной кибернетики» на тему «Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях». М.: НИТУ «МИСИС», 2025. С. 35-42.

[3] Дорошев П.И., Хижняк М.А. Исследование возможности распознавания и классификации галактик на астрономических снимках с помощью методов компьютерного зрения // Сборник статей научно-технического семинара кафедры «Инженерной кибернетики» на тему «Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях». М.: НИТУ «МИСИС», 2025. С. 50-56.

[4] Ali B., Sadekov R.N., Tsodokova V.V. A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems // Gyroscopy and Navigation. 2022. Vol. 13. P. 241-252. DOI: 10.1134/S2075108722040022.

[5] Pazychev D., Bakulev K., Sadekov R. Low-Cost Navigation System for UAV // Proceedings of the International Conference on Intelligent Navigation Systems (ICINS). 2023. P. 1-6. DOI: 10.23919/ICINS51816.2023.10168469.

[6] Krizhevsky A., Sutskever I., Hinton G.E. ImageNet classification with deep convolutional neural networks // Advances in neural information processing systems. 2012. Vol. 25. P. 1097-1105.

[7] Hochreiter S., Schmidhuber J. Long short-term memory // Neural computation. 1997. Vol. 9. No. 8. P. 1735-1780.

[8] Cao Z. et al. OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields // IEEE transactions on pattern analysis and machine intelligence. 2019. Vol. 43. No. 1. P. 172-186.

[9] Lugaresi C. et al. MediaPipe: A framework for building perception pipelines // arXiv preprint arXiv:1906.08172. 2019.

[10] Cortes C., Vapnik V. Support-vector networks // Machine learning. 1995. Vol. 20. P. 273-297.

[11] Hand Gestures Recognition Dataset [Электронный ресурс]. URL: https://huggingface.co/datasets/wonderize/cv-gesture-recognizer-paper (дата обращения: 04.01.2026).

[12] LeCun Y., Bengio Y., Hinton G. Deep learning // Nature. 2015. Vol. 521. No. 7553. P. 436-444.

[13] Abadi M. et al. TensorFlow: Large-scale machine learning on heterogeneous systems // Software available from tensorflow.org. 2015.

ГРНТИ 28.23.15
ASJC 1702, 1707
OECD 01.02.EP

Определение художественного стиля картины по изображению на основе нейросетевых моделей

А. В. Васильчиков
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2108434@edu.misis.ru

***Аннотация* — рост объёмов цифровых художественных архивов повышает потребность в инструментах, позволяющих определять художественный стиль и ускорять каталогизацию музейных коллекций и онлайн-галерей. В статье решается задача определения художественного стиля картины по изображению на пользовательском наборе данных, сформированном из открытых источников и включающем 11 стилей. Для решения применены современные нейросетевые архитектуры Vision Transformer, Swin Transformer и ConvNeXt. Результаты показывают, что наилучшее качество классификации художественных стилей достигается при использовании Swin Transformer, что подтверждает применимость подхода для автоматизации атрибуции, построения тематических подборок и поиска стилистически близких произведений.**

Ключевые слова — компьютерное зрение, классификация изображений, художественный стиль, живопись, нейронные сети, трансформерные архитектуры, сверточные нейронные сети, анализ ошибок классификации.

I. Введение

Цифровизация музейных и частных коллекций, развитие онлайн-галерей и рекомендательных сервисов приводят к росту объёма изображений произведений живописи, которые необходимо каталогизировать и индексировать по содержательным признакам. Одним из наиболее значимых атрибутов является художественный стиль, так как он используется при построении тематических подборок, навигации по коллекции и поиске стилистически близких работ. Ручная обработка требует участия экспертов и плохо масштабируются при больших объёмах данных, поэтому возникает потребность в автоматизированном определении художественного стиля по изображению. Для решения подобных задач применяются методы компьютерного зрения и нейросетевой анализ изображений [1].

Современные методы компьютерного зрения и глубокого обучения показали высокую эффективность в задачах классификации изображений в различных прикладных областях, где также присутствует существенная вариативность данных и высокие требования к обобщающей способности моделей. В частности, нейросетевые подходы успешно применяются для анализа медицинских изображений, где требуется выявление значимых визуальных закономерностей и последующая классификация состояния по изображению [2]. Аналогично, в задачах компьютерного зрения для автономных систем подчёркивается роль устойчивых алгоритмов распознавания и интерпретации визуальной информации как основы для дальнейших прикладных решений [3-4]. Эти примеры демонстрируют, что перенос современных моделей на художественные изображения является обоснованным, однако требует корректной постановки эксперимента и оценки качества.

Задача определения художественного стиля относится к задачам классификации изображений и обладает рядом особенностей. Разные направления живописи могут быть близки по композиционным и колористическим приёмам, а в рамках одного направления встречается высокая вариативность, связанная с сюжетами, техникой исполнения и качеством оцифровки. Поэтому важно сравнивать архитектуры, способные учитывать как локальные признаки (текстуры, мазки), так и глобальные зависимости (композиция, структура сцены). В последние годы для задач классификации изображений широкое распространение получили трансформерные модели, включая Vision Transformer [5] и Swin Transformer [6]. Параллельно развиваются современные сверточные архитектуры нового поколения, такие как ConvNeXt, которые сохраняют эффективность сверточных сетей и демонстрируют конкурентоспособность по качеству [7].

Отдельного внимания требует набор данных, поскольку достоверность выводов о качестве моделей зависит от согласованности разметки, отсутствия утечек между выборками и корректного разбиения на обучающую, валидационную и тестовую части. Практика прикладных исследований показывает, что качество подготовки данных и единый протокол сравнения определяют воспроизводимость результатов и корректность интерпретации метрик [8].

В связи с этим целью настоящей работы является исследование возможности определения художественного стиля картин на пользовательском наборе данных и сопоставление современных нейросетевых архитектур по единому протоколу обучения и оценки.

II. Наборы данных

Для проведения экспериментов был сформирован пользовательский набор изображений живописи, предназначенный для задачи определения художественного стиля по изображению. В качестве исходного материала использовались открытые изображения из коллекции WikiArt, а также открытые наборы данных, представленные на платформе Kaggle,

после чего данные были отфильтрованы и приведены к единому формату разметки. Итоговый набор содержит 11223 изображения и охватывает 11 художественных стилей: Abstract Expressionism, Baroque, Cubism, Expressionism, High Renaissance, Impressionism, Post Impressionism, Realism, Rococo, Romanticism, Symbolism. Для иллюстрации визуального разнообразия и различий между стилями на рисунке 1 приведены примеры изображений из каждого стиля.

Рис. 1. Примеры картин каждого стиля

Разметка датасета представлена в виде метки класса для каждого изображения. При подготовке набора данных обеспечивалась согласованность структуры, исключение дублирующихся экземпляров и корректность сопоставления изображения и метки. Для удобства воспроизведения экспериментов и дальнейшего использования данные были организованы в иерархической структуре каталогов и дополнены метаданными. Сформированный набор данных был опубликован в открытом доступе в репозитории Hugging Face Datasets, что обеспечивает возможность проверки результатов и повторения экспериментов [9].

Набор данных был разделён на три непересекающиеся части: обучающую, валидационную и тестовую. Обучающая выборка использовалась для оптимизации параметров нейросетевых моделей. Валидационная выборка применялась для подбора гиперпараметров и контроля переобучения в ходе обучения. Тестовая выборка предназначалась для итоговой независимой оценки качества выбранной модели. Разделение выполнялось со стратификацией по художественным стилям, что позволило сохранить сопоставимое распределение стилей во всех подвыборках и обеспечить корректность сравнения моделей.

Дополнительно для обеспечения воспроизводимости был сформирован файл метаданных, связывающий изображения, классы и принадлежность к одной из трех выборок (обучающей, валидационной или тестовой), а также подготовлена разметка в формате COCO для каждой части набора данных. Опубликованный набор данных может использоваться для последующего сравнения моделей и дальнейших исследований в области анализа художественных изображений.

III. Нейросетевые архитектуры

Для решения задачи определения художественного стиля по изображению были рассмотрены три современные архитектуры, широко применяемые в классификации изображений и демонстрирующие высокие результаты при переносе обучения на прикладные наборы данных: Vision Transformer, Swin Transformer и ConvNeXt. Выбор этих моделей обусловлен тем, что они представляют разные подходы к извлечению визуальных признаков: трансформер с глобальным вниманием, иерархический трансформер с локальным вниманием и «современную» сверточную сеть. Такой набор позволяет сравнить, какие индуктивные предпосылки (глобальный контекст, локальные окна внимания, свертки) оказываются более полезными для распознавания художественного стиля.

A. Vision Transformer

Vision Transformer (ViT) переносит идею трансформеров на изображения, представляя входное изображение как последовательность патчей фиксированного размера. Каждый патч преобразуется в эмбеддинг, к последовательности добавляются позиционные представления, а далее применяется стандартный трансформерный энкодер с механизмом самовнимания, который моделирует зависимости между всеми патчами изображения. Это позволяет учитывать глобальные отношения в композиции и взаимосвязи между удалёнными областями кадра. Подход был показан как конкурентоспособный по качеству при условии предварительного обучения на больших наборах данных и последующей донастройки под целевую задачу [5]. Принцип работы ViT наглядно представлен на рисунке 2.

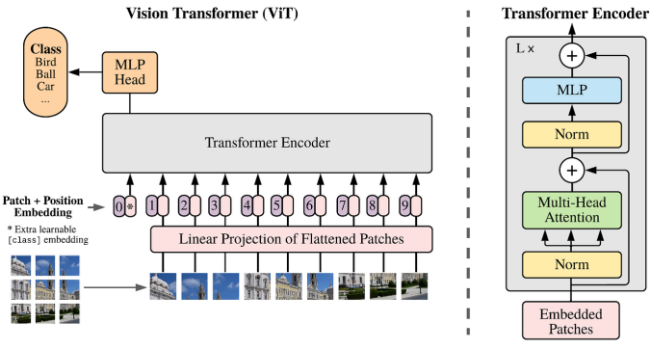


Рис. 2 Принцип работы ViT

К преимуществам ViT обычно относят способность эффективно использовать глобальный контекст изображения и хорошую масштабируемость по размеру модели и объёму данных. Ограничением является более слабая «врождённая» ориентация на локальные текстуры по сравнению со сверточными сетями, а также чувствительность к объёму обучающих данных при обучении «с нуля». В рамках данной работы ViT рассматривался как базовая трансформерная модель для сопоставления с более структурированными архитектурами.

B. Swin Transformer

Другой Swin Transformer развивает идею трансформеров для компьютерного зрения за счёт иерархической обработки признаков и ограничения самовнимания локальными окнами.

На ранних стадиях модель работает с патчами высокой детализации, затем последовательно выполняется объединение патчей (уменьшение пространственного разрешения при росте размерности признаков), что формирует многоуровневое представление, характерное для классических сверточных сетей. Ключевая особенность Swin Transformer - смещение окон внимания между соседними блоками, благодаря чему обеспечиваются связи между окнами и лучше учитывается контекст за пределами локального региона при сохранении вычислительной эффективности [6]. На рисунке 3 представлена архитектура Swin Transformer.

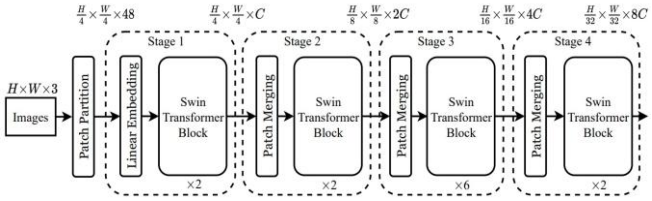


Рис. 3 Архитектура Swin Transformer

Для задачи художественного стиля Swin Transformer особенно интересен тем, что сочетает локальное внимание (полезное для мазков, текстур и мелких элементов) и иерархическое укрупнение признаков (полезное для композиции и глобальной структуры). Кроме того, вычислительная сложность внимания становится более управляемой на изображениях стандартного размера. Поэтому данная архитектура рассматривалась как основной кандидат на роль наиболее устойчивой и точной модели в условиях ограниченного объёма целевого набора данных.

C. ConvNeXt

ConvNeXt относится к архитектурам «нового поколения» сверточных сетей. Идея модели состоит в том, чтобы сохранить простоту и сильные стороны сверток (локальность, устойчивость к небольшим деформациям и текстурная чувствительность), но модернизировать дизайн по аналогии с успешными практиками трансформеров. В ConvNeXt используются современные решения на уровне блоков и стадий сети, такие как увеличенные ядра глубинных сверток, изменённые нормализации и перераспределение вычислений между стадиями. Авторы показывают, что при такой модернизации чисто сверточная архитектура способна конкурировать с трансформерами по качеству на задачах распознавания и ряде других задач компьютерного зрения [7]. На рисунке 4 представлена упрощенная схема блока ConvNeXt.

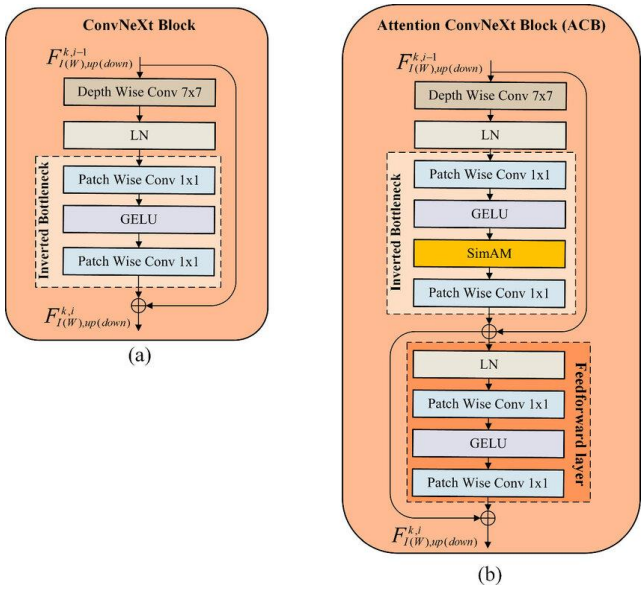


Рис. 4 Упрощенная схема блока ConvNeXt

На рисунке 4 блок состоит из глубинной свёртки с увеличенным размером ядра, за которой следует нормализация признаков и последовательность точечных свёрток, реализующих нелинейное преобразование признакового пространства. Использование остаточного соединения позволяет стабилизировать процесс обучения и способствует более эффективному распространению градиентов в глубокой сети. Такая структура блока сохраняет ключевые свойства сверточных нейронных сетей, включая локальность обработки и устойчивость к небольшим деформациям изображения, одновременно заимствуя ряд архитектурных решений из трансформерных моделей. В результате ConvNeXt демонстрирует конкурентоспособность по качеству на задачах классификации изображений, оставаясь при этом вычислительно эффективной и хорошо интерпретируемой архитектурой.

Для распознавания художественного стиля ConvNeXt является важным участником сравнения, поскольку стиль часто проявляется через локальные признаки: характер мазка, зернистость, особенности линий и мелких деталей. Сверточные сети традиционно сильны в таких паттернах, поэтому ConvNeXt позволяет проверить, насколько современные сверточные решения сопоставимы с трансформерами на художественных изображениях.

D. Обоснование выбора моделей

Таким образом, выбранные архитектуры формируют уникальные подходы к извлечению и анализу признаков:

- ViT акцентирует глобальные зависимости между частями изображения за счёт полного самовнимания [5];
- Swin Transformer сочетает локальность и иерархию представлений, расширяя контекст за счёт смещения окон внимания [6];
- ConvNeXt сохраняет сверточные индуктивные предпосылки, но использует современный дизайн, приближаясь по качеству к трансформерам [7].

Сопоставление моделей выполняется в едином протоколе обучения и оценки на одном пользовательском наборе данных, что обеспечивает корректность сравнения и интерпретацию различий в качестве.

IV. Сравнение моделей

Для выбора базовой архитектуры была проведена серия экспериментов по единому протоколу обучения и оценки для трёх моделей: Vision Transformer (ViT-B/16), Swin Transformer (Swin-Base) и ConvNeXt (ConvNeXt-Base). Во всех экспериментах использовался один и тот же пользовательский набор данных, одинаковое разрешение входных изображений и единая схема разбиения на обучающую, валидационную и тестовую выборки. В процессе обучения изображения масштабировались до фиксированного разрешения, принятого в используемых моделях, что обеспечивало корректную подачу данных в нейросетевые архитектуры и сопоставимость результатов.

Для количественной оценки качества моделей использовались метрики accuracy и macro-F1, позволяющие охарактеризовать как общую долю верных предсказаний, так и устойчивость распознавания по всем художественным стилям

Accuracy (доля верных предсказаний) отражает отношение числа правильно классифицированных изображений к общему числу изображений в выборке. При наличии N объектов и $N_{correct}$ верных ответов значение метрики вычисляется по формуле:

$$\text{Accuracy} = \frac{N_{\text{correct}}}{N}, \quad (1)$$

где $N_{correct}$ – количество верных ответов, N – количество всех ответов.

В терминах матрицы ошибок C (confusion matrix) для задачи с K классами accuracy равна отношению суммы элементов главной диагонали к сумме всех элементов матрицы к сумме всех элементов матрицы:

$$\text{Accuracy} = \frac{\sum_{i=1}^{K} C_{ii}}{\sum_{i=1}^{K} \sum_{j=1}^{K} C_{ij}}, \quad (2)$$

где K – количество классов, C – матрица ошибок, C_{ij} - число объектов, которые в действительности принадлежат классу i, но модель предсказала класс j, C_{ii} - число верно классифицированных объектов класса i.

Данная метрика удобна для общей интерпретации качества, однако в задачах многоклассовой классификации может скрывать существенные различия в качестве распознавания отдельных стилей.

Для более детальной оценки по каждому стилю применяются показатели: точность (precision), полнота (recall), F1, macro-F1. Они определяются как:

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

где TP - число истинно-положительных предсказаний, FP – число ложноположительных предсказаний.

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

где FN - число ложноотрицательных предсказаний.

- $$F1 = 2\frac{Precision \cdot Recall}{Precision+Recall} = \frac{2TP}{2TP+FP+FN} \quad (5)$$

где F1 - гармоническое среднее между precision и recall, отражающее баланс двух метрик.

- $$\text{Macro-F1} = \frac{1}{K}\sum_{i=1}^{K} F\,1_i \quad (6)$$

где K – число классов, Macro-F1 - среднее значение F1 по всем классам, где каждый стиль имеет одинаковый вес.

Использование macro-F1 позволяет корректнее отражать качество распознавания в многоклассовой постановке, так как метрика учитывает эффективность модели для каждого стиля отдельно и снижает влияние ситуаций, когда рост accuracy обеспечивается преимущественно за счёт более простых для распознавания классов.

Оценка проводилась на валидационной и тестовой выборках: валидационная применялась для контроля динамики обучения и выбора наилучшей конфигурации модели, а тестовая — для итоговой независимой оценки качества. Для корректного сопоставления архитектур основное внимание уделялось результатам на тестовой выборке, поскольку именно они характеризуют способность модели обобщать на данных, не участвовавших ни в обучении, ни в подборе параметров.

Обучение всех моделей проводилось по единому протоколу, чтобы обеспечить корректность сравнения архитектур. В качестве критерия оптимизации использовалась кросс-энтропийная функция потерь для многоклассовой классификации, так как она напрямую соответствует задаче выбора одного стиля из фиксированного набора и является стандартом для подобных постановок. Оптимизация выполнялась методом AdamW с регуляризацией по весам (weight decay), что позволяет снизить склонность модели к переобучению при дообучении предобученных сетей на относительно небольшом пользовательском наборе данных.

Число эпох ограничивалось небольшим диапазоном (5 эпохами), поскольку все рассматриваемые архитектуры дообучались от предобученных весов, а при такой схеме основной прирост качества обычно достигается в первые несколько эпох. Далее при увеличении числа эпох заметно возрастает риск переобучения, что проявляется в расхождении динамики обучающей и валидационной функций потерь. Поэтому контроль качества выполнялся по валидационной выборке, а сохранение модели осуществлялось по лучшему значению выбранной метрики на валидации (в экспериментах ориентировались на максимизацию accuracy при обязательном контроле macro-F1). Такой подход позволяет фиксировать состояние модели, обеспечивающее наилучший компромисс между общим качеством распознавания и равномерностью по стилям.

Дополнительно в экспериментах применялась разминка скорости обучения (warmup) в начале оптимизации и обучение со смешанной точностью (fp16) при наличии графического ускорителя. Разминка задавалась параметром warmup_ratio и позволяла сделать старт обучения более стабильным. Использование fp16 уменьшало нагрузку на видеопамять и ускоряло вычисления на GPU. Размер пакета подбирался с учётом доступной памяти, а при необходимости применялось накопление градиентов, что позволяло сохранить заданный эффективный размер пакета при меньшем batch size на устройство.

По результатам обучения всех трех моделей сформирована таблица I.

ТАБЛИЦА I. Результаты обучения базовых моделей на пользовательском датасете

	Swin-Base	**ViT-B/16**	**ConvNeXt-Base**
Accuracy (val)	0.718	0.679	0.647
Macro-F1 (val)	0.716	0.675	0.644
Loss (val)	0.856	0.974	0.990
Accuracy (test)	0.716	0.680	0.657
Macro-F1 (test)	0.716	0.676	0.655
Loss (test)	0.843	0.963	0.986

Из представленной таблицы, видно, что среди рассмотренных архитектур наилучшее качество демонстрирует Swin Transformer. Его преимущество проявляется и в точности, и в значении macro-F1, что указывает на более стабильную работу по стилям в целом. Vision Transformer занимает промежуточное положение: качество заметно выше, чем у ConvNeXt, однако уступает Swin. ConvNeXt в выбранных условиях обучения показал наименьшие значения метрик, что может быть связано как с особенностями архитектуры, так и с чувствительностью к настройкам обучения и аугментациям для художественных изображений.

V. Дообучение выбранной модели

По результатам базового сравнения архитектур в качестве основной модели для дальнейшего дообучения был выбран Swin Transformer, показавший наилучший баланс качества по метрикам accuracy и macro-F1 на отложенных данных. На следующем этапе выполнялась донастройка модели на пользовательском датасете с сохранением единого протокола разбиения данных.

Дообучение выбранной модели выполнялось от сохранённой лучшей версии с адаптацией предобученных весов под пользовательский датасет. Обучение проводилось до 8 эпох при сниженной скорости обучения $2 \cdot 10^{-5}$. Размер пакета на обучение (и оценку) составлял 16 изображений на устройство, при этом применялось накопление градиентов в течение 2 шагов, что обеспечивало эффективный размер пакета 32.

Контроль качества выполнялся по валидационной выборке после каждой эпохи, лучшая версия модели выбиралась по метрике accuracy на валидации, а для предотвращения переобучения использовалась ранняя остановка с параметром patience = 2. При наличии графического ускорителя применялся режим смешанной точности fp16, ускоряющий вычисления и снижающий потребление видеопамяти.

Ход дообучения анализировался по динамике функции потерь и метрик качества на валидационной выборке. На рисунке 5 приведена динамика валидационной accuracy и macro-F1 по эпохам, что позволяет визуально оценить, на каком интервале обучение стабилизируется и где достигается максимум качества.

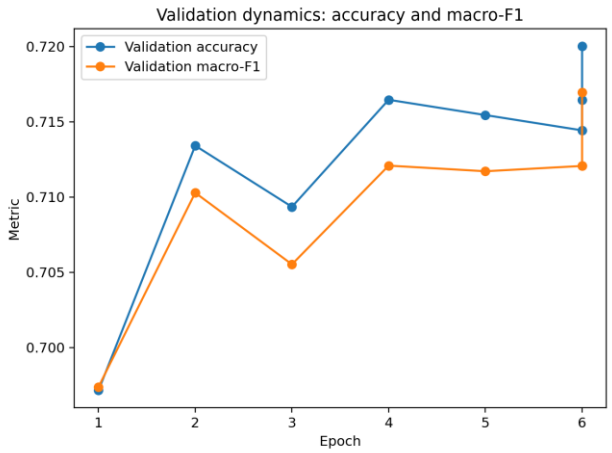


Рис. 5 Динамика accuracy и macro-F1 на валидационной выборке по эпохам

На рисунке 6 показаны кривые training loss и validation loss, используемые для контроля возможного переобучения: устойчивое снижение training loss при стагнации или росте validation loss может указывать на ухудшение обобщающей способности.

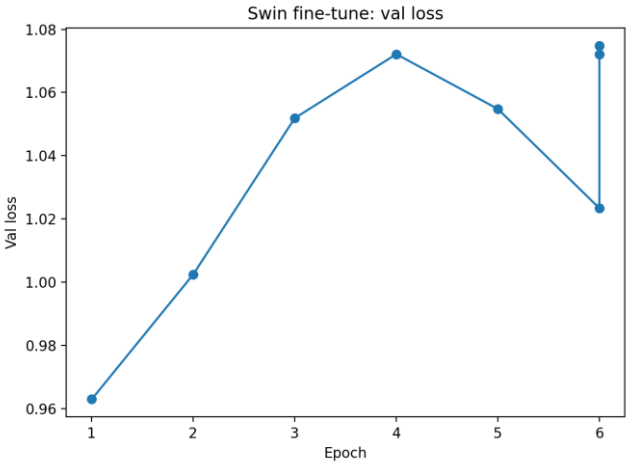
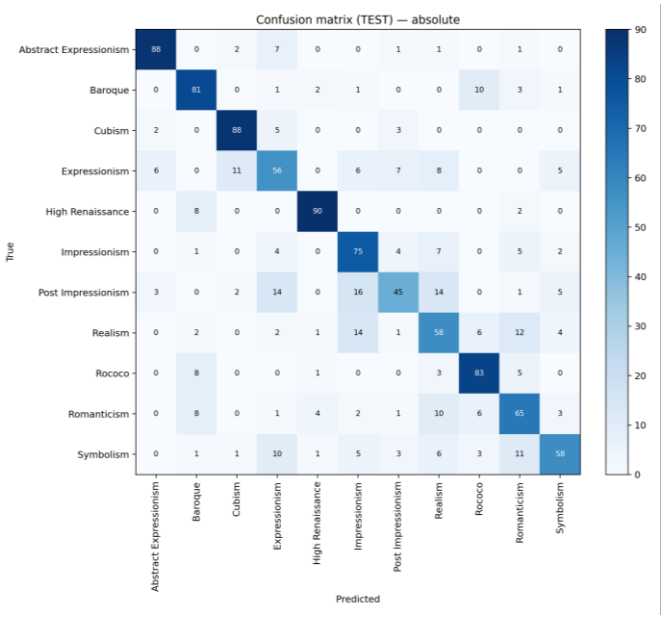


Рис. 6 Динамика training и validation loss при дообучении модели

Лучшая версия модели была зафиксирована на 6-й эпохе обучения. Она определялась по максимальному значению accuracy на валидационной выборке (при контроле macro-F1): после 6-й эпохи дальнейшее обучение не давало устойчивого улучшения на валидации и сопровождалось признаками начинающегося переобучения (training loss продолжал снижаться, тогда как validation loss и метрики качества переставали улучшаться). Поэтому для итоговой оценки на тестовой выборке использовалась модель, сохранённая после 6-й эпохи как лучшая по валидации.

После выбора лучшей версии по валидационной выборке модель оценивалась на тестовой части. Для интерпретации результатов рассматривались матрицы ошибок (рисунок 7). Ненормализованная отражает абсолютное число ошибок между стилями, а нормализованная сравнивает структуру ошибок независимо от числа примеров. Кроме того, для полноты оценки приведен отчёт значения точности, полноты и F1-score при одинаковом объёме примеров в классах (табл. II).



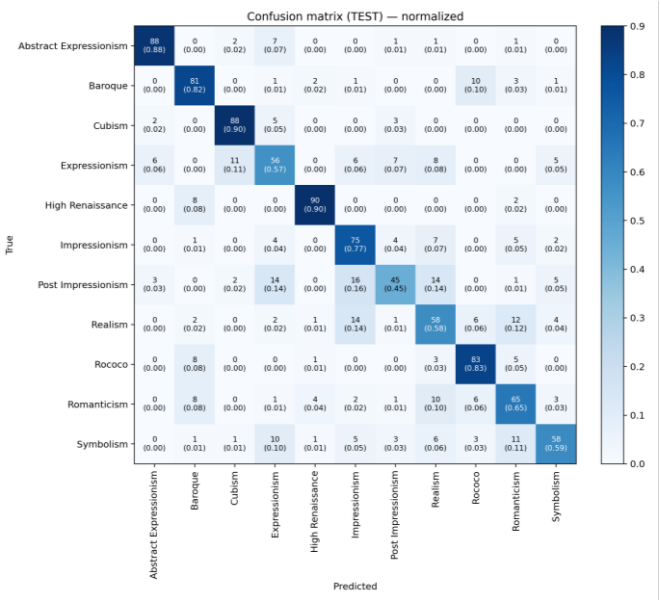


Рис. 7 Матрицы ошибок (абсолютная и нормализированная) для итоговой модели

ТАБЛИЦА II. Результаты классификации по стилям на тестовой выборке для дообученной модели

Стиль	Precision	Recall	F1-score	Support
Abstract Expressionism	0.8889	0.8800	0.8844	100
Baroque	0.7431	0.8182	0.7788	99
Cubism	0.8462	0.8980	0.8713	98
Expressionism	0.5600	0.5657	0.5628	99
High Renaissance	0.9091	0.9000	0.9045	100
Impressionism	0.6303	0.7653	0.6912	98
Post Impressionism	0.6923	0.4500	0.5455	100
Realism	0.5421	0.5800	0.5604	100
Rococo	0.7685	0.8300	0.7981	100
Romanticism	0.6190	0.6500	0.6341	100
Symbolism	0.7436	0.5859	0.6554	99

Анализ матрицы ошибок и представленной таблицы, показывает, что по стилям наиболее устойчиво распознаются направления с выраженными визуальными маркерами. На тестовой выборке наивысшие значения F1-score получены для High Renaissance (0,904), Abstract Expressionism (0,884) и Cubism (0,871). Это согласуется с тем, что для этих направлений характерны относительно стабильные признаки композиции и структуры изображения, либо специфическая фактура и манера письма. Наиболее сложными остаются Post Impressionism (F1 0,546 при recall 0,45), Expressionism (F1 0,563) и Realism (F1 0,560), что связано с высокой вариативностью внутри направлений и частичным пересечением визуальных приёмов.

Структура ошибок по матрице показывает, что перепутывания возникают в основном между близкими стилями. Наиболее заметны ошибки между Post Impressionism и Impressionism (16%), а также между Realism и Impressionism (14%), что объясняется сходством колористики и манеры письма у части работ, а также влиянием условий оцифровки. Для Expressionism характерны ошибки в сторону Cubism (11%), что может быть связано с близкими приёмами деформации формы и упрощения пространственной структуры. Также выделяются взаимные перепутывания между Symbolism и Romanticism (11%), Rococo и Baroque (8–10%), что отражает историческую и визуальную близость этих направлений. Это согласуется с тем, что стилевые различия в живописи часто определяются одновременно и локальными деталями (характер мазка, текстуры), и глобальными зависимостями (композиция), поэтому архитектуры, учитывающие оба типа признаков, демонстрируют более стабильное качество.

Из представленной видно, что наилучшие значения F1-score получены для High Renaissance, Abstract Expressionism и Cubism, что указывает на наличие устойчивых визуальных признаков в этих стилях. Наиболее проблемными остаются Post Impressionism, Expressionism и Realism, причём для Post Impressionism наблюдается заметно сниженная полнота (recall), то есть часть работ данного направления чаще относится к близким стилям. В целом распределение метрик по стилям подтверждает, что основные ошибки связаны с визуально сходными направлениями, поэтому дальнейшее улучшение качества целесообразно направлять на уточнение границ между близкими стилями и анализ наиболее конфликтных пар по матрице ошибок.

Наконец, рассмотрим сравнительный результат дообучения модели. Для этого была составлена таблица III.

ТАБЛИЦА III. Итоговые метрики качества для базовой и дообученной версии Swin Transformer

	Swin-Base (5 ep)	Swin-Base + fine-tune (best)
Accuracy	0,716	0,72
Macro-F1	0,716	0,717
Loss	0,843	1,075

Анализируя представленную таблицу, видно, что при дообучении лучшей модели наблюдается рост значения функции потерь при одновременном небольшом повышении accuracy и macro-F1. Такая ситуация возможна, поскольку кросс-энтропийная функция потерь учитывает не только факт правильной классификации, но и уверенность модели в предсказании. В ходе доработки распределения вероятностей модель может чаще выбирать верный стиль, однако отдельные сложные изображения она может классифицировать ошибочно с высокой уверенностью, что приводит к увеличению среднего значения loss. В связи с этим при итоговой интерпретации результатов приоритетными показателями качества рассматриваются accuracy и macro-F1, а loss используется как дополнительный индикатор поведения оптимизации.

В целом проведенное сравнение и последующая доработка модели показывают, что использование архитектуры Swin Transformer обеспечивает наиболее устойчивое и сбалансированное качество распознавания художественных стилей на пользовательском наборе данных. Анализ интегральных метрик и структуры

ошибок подтверждает, что основные трудности связаны с визуально близкими направлениями живописи, тогда как для стилей с выраженными композиционными и текстурными признаками достигается высокая точность. Полученные результаты обосновывают выбор дообученной версии модели в качестве итоговой и создают основу для дальнейших исследований, направленных на снижение ошибок между близкими художественными стилями.

VI. Заключение

В ходе работы была рассмотрена задача определения художественного стиля картин по изображению с использованием методов компьютерного зрения и современных нейросетевых архитектур. В ходе исследования был сформирован и опубликован пользовательский набор данных, включающий изображения одиннадцати художественных стилей, полученные из открытых источников и приведённые к единому формату разметки. Корректная подготовка датасета и разделение на обучающую, валидационную и тестовую выборки позволили обеспечить воспроизводимость экспериментов и корректность сравнения моделей.

В работе выполнено сравнение трёх современных архитектур классификации изображений, а именно Vision Transformer, Swin Transformer и ConvNeXt, при использовании единого протокола обучения и оценки. Результаты показали, что на рассматриваемом наборе данных наилучшее качество демонстрирует архитектура Swin Transformer. Она обеспечивает более высокий и устойчивый уровень точности и значения macro-F1 по сравнению с другими моделями. Это указывает на эффективность иерархического трансформерного подхода в задачах, где требуется учитывать как локальные текстурные признаки, так и глобальные особенности композиции изображения.

Для выбранной архитектуры было проведено дополнительное дообучение с использованием предобученных весов. Анализ динамики обучения и результатов на отложенной тестовой выборке показал, что fine-tuning позволяет добиться умеренного улучшения качества распознавания при сохранении устойчивости по стилям. Детальный анализ по классам выявил, что наибольшие трудности возникают при различении художественных направлений с близкими визуальными характеристиками, таких как импрессионизм и постимпрессионизм или барокко и рококо. В то же время стили с выраженными композиционными и текстурными признаками распознаются более надёжно.

Полученные результаты подтверждают применимость современных нейросетевых моделей для автоматического определения художественного стиля картин. Работа показывает важность комплексной оценки качества, включающей анализ интегральных метрик и структуры ошибок по стилям. Сформированный пользовательский датасет и проведённые эксперименты могут служить основой для дальнейших исследований, направленных на расширение набора художественных стилей, повышение точности распознавания визуально близких направлений и разработку прикладных систем анализа изображений произведений живописи.

Литература

[1] Кудинов, Я. О. Исследование возможности классификации картин при помощи компьютерного зрения / Я. О. Кудинов // 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. 106-111. – EDN QGVKPE.

[2] Темкин, И. О. Нейросетевая модель распознавания патологий глазного дна на основе оптической когерентной томографии / И. О. Темкин, В. В. Горин // Вестник Череповецкого государственного университета. – 2025. – № 1(124). – С. 70-79. – DOI 10.23859/1994-0637-2025-1-124-6. – EDN EBEOLO.

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

[4] Guzhva, Nikita & Sadekov, Rinat & Ali, Bushra & Sholokhov, A & Bakulev, K. (2023). Evaluating the Accuracy of Tram Positioning System in High-Rise Building Environment Using Data from Visual Geoinformation Systems. 10.23919/ICINS51816.2023.10168407.

[5] Dosovitskiy, Alexey & Beyer, Lucas & Kolesnikov, Alexander & Weissenborn, Dirk & Zhai, Xiaohua & Unterthiner, Thomas & Dehghani, Mostafa & Minderer, Matthias & Heigold, Georg & Gelly, Sylvain & Uszkoreit, Jakob & Houlsby, Neil. (2020). An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. 10.48550/arXiv.2010.11929.

[6] Liu, Ze & Lin, Yutong & Cao, Yue & Hu, Han & Wei, Yixuan & Zhang, Zheng & Lin, Stephen & Guo, Baining. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. 9992-10002. 10.1109/ICCV48922.2021.00986.

[7] Liu, Zhuang & Mao, Hanzi & Wu, Chao-Yuan & Feichtenhofer, Christoph & Darrell, Trevor & Xie, Saining. (2022). A ConvNet for the 2020s. 11966-11976. 10.1109/CVPR52688.2022.01167.

[8] 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.

[9] Huggingface—Painting_Styles. —2025— Available at: https://huggingface.co/datasets/VasilchikovAV/Painting_Styles/tree/main (Accessed: 27.12.2025).

[10] Wolf, Thomas & Debut, Lysandre & Sanh, Victor & Chaumond, Julien & Delangue, Clement & Moi, Anthony & Cistac, Pierric & Rault, Tim & Louf, Remi & Funtowicz, Morgan & Davison, Joe & Shleifer, Sam & Platen, Patrick & Ma, Clara & Jernite, Yacine & Plu, Julien & Xu, Canwen & Scao, Teven & Gugger, Sylvain & Rush, Alexander. (2020). Transformers: State-of-the-Art Natural Language Processing. 38-45. 10.18653/v1/2020.emnlp-demos.6.

[11] Gatys, Leon & Ecker, Alexander & Bethge, Matthias. (2015). A Neural Algorithm of Artistic Style. arXiv. 10.1167/16.12.326.

ГРНТИ 28.23.13, 28.23.15, 44.47.01
ASJC 1702, 1710
OECD 1.02.EP

Нейросетевые методы для определения внешнего состояния автомобиля

М. С. Гришечкин
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2508140@edu.misis.ru

М. Л. Сульповар
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2508257@edu.misis.ru

***Аннотация* — в условиях цифровизации транспортной отрасли и роста объёма визуальных данных всё более актуальной становится задача автоматизированной оценки внешнего состояния автомобиля. Такая оценка необходимо при технической диагностике, страховом осмотре, мониторинге автопарков и предпродажной экспертизе. Традиционные методы визуального контроля требуют значительных трудозатрат и подвержены субъективным ошибкам. В последние годы методы глубокого обучения и компьютерного зрения продемонстрировали высокую эффективность при решении задач обнаружения и классификации визуальных дефектов. В данной работе рассматриваются нейросетевые методы распознавания повреждений автомобиля, таких как трещины, вмятины и сломанные элементы кузова, на основе изображений. Для решения поставленной задачи используются современные модели с открытым исходным кодом Grounding DINO и OWLv2, ориентированные на детекцию объектов и визуально-языковое сопоставление. Проводится анализ их возможностей по обнаружению и локализации различных типов повреждений на изображениях автомобилей. Полученные результаты демонстрируют перспективность применения универсальных моделей детекции для автоматизированной оценки внешнего состояния транспортных средств и подтверждают их практическую значимость для задач технической диагностики и страховой экспертизы.**

Ключевые слова — автоматизированная диагностика внешнее состояние автомобиля, компьютерное зрение, нейронные сети, обнаружение дефектов, оценка повреждений, распознавание повреждений, техническое состояние транспортных средств, цифровая обработка изображений.

I. Введение

Автоматизация процессов технической диагностики и контроля состояния транспортных средств является одной из актуальных задач современной автомобильной отрасли. Оценка внешнего состояния автомобиля востребована при страховом осмотре, техническом обслуживании, мониторинге автопарков, а также при купле-продаже транспортных средств. Традиционный визуальный осмотр, выполняемый специалистом, требует значительных временных затрат и подвержен субъективным факторам, что обуславливает необходимость разработки автоматизированных систем анализа визуальных данных [1]. В связи с этим все более широкое применение находят методы компьютерного зрения, позволяющие анализировать изображения автомобиля и выявлять внешние повреждения, такие как трещины, вмятины и сломанные элементы кузова.

Задача распознавания повреждений автомобиля по изображениям относится к классу задач обнаружения и классификации объектов и характеризуется высокой сложностью из-за разнообразия форм дефектов, условий освещения, ракурсов съемки и особенностей фона. Для ее решения активно применяются методы глубокого обучения, продемонстрировавшие высокую эффективность при анализе визуальных данных [2, 3]. Современные нейросетевые модели способны не только обнаруживать объекты на изображении, но и обобщать знания на ранее не встречавшиеся категории, что особенно важно при ограниченном объеме аннотированных данных [4].

Особый интерес представляют универсальные модели детекции, основанные на визуально-языковых представлениях, которые позволяют выполнять обнаружение объектов на основе текстовых описаний. К таким моделям относятся Grounding DINO и OWLv2, использующие механизмы совместного анализа визуальной и текстовой информации [5]. Применение подобных подходов снижает зависимость от фиксированного набора классов и расширяет возможности использования нейросетевых методов в задачах распознавания повреждений автомобиля. В данной работе рассматриваются и анализируются возможности применения указанных моделей для автоматизированного обнаружения и локализации внешних дефектов транспортных средств по изображениям, а также оценивается их практическая применимость в задачах технической диагностики.

II. Наборы данных

Для обучения и оценки нейросетевых моделей был сформирован специализированный датасет изображений автомобилей с аннотированными внешними повреждениями, ориентированный на решение задачи обнаружения и локализации дефектов кузова и внешних элементов автомобиля. В качестве основы для формирования датасета был использован открытый набор данных CarDD (Car Damage Dataset), широко применяемый в задачах детекции и классификации повреждений автомобилей [6]. Использование данного набора данных позволило обеспечить релевантность изображений и классов повреждений в контексте поставленной задачи.

Разметка изображений была выполнена вручную с использованием инструмента CVAT.ai. В процессе

разметки особое внимание уделялось точной локализации повреждений и единообразию классов, что позволило повысить качество обучающих данных.

Рис. 1. Примеры аннотированных изображений в CVAT.ai

В рамках разметки были выделены следующие классы повреждений:

- **dent** – вмятина;
- **scratch** – царапина;
- **rust** – коррозионное повреждение;
- **wheel damage** – повреждение колеса;
- **fracture** – трещина/разбитое стекло;
- **broken** – полностью сломанный кузовной элемент;
- **misaligned** – смещенный/погнутый элемент кузова.

Сформированный датасет включает около 1097 изображений автомобилей, полученных при различных условиях освещения, с разнообразными ракурсами съемки и фоном. В выборку включены изображения, снятые как в контролируемых, так и в сложных условиях, включая наличие теней, бликов, частичное перекрытие элементов автомобиля и визуальные помехи, что отражает реальные сценарии эксплуатации транспортных средств.

Рис. 2. Примеры изображений из сформированного датасета

Дополнительно в датасет были включены сложные случаи, слабо представленные в исходных открытых источникам, в том числе сцены с несколькими повреждениями на одном автомобиле, а также мелкие и слабо выраженные повреждения. Такой подход позволяет повысить устойчивость обучаемых моделей к вариативности визуальных признаков и улучшить их обобщающую способность.

Рис. 3. Примеры кадров со сложными и малозаметными повреждениями

Также обеспечена сбалансированность классов повреждений: каждое изображение содержит как минимум один экземпляр одного из целевых классов, что позволило избежать смещения выборки в сторону отдельных типов дефектов и обеспечить репрезентативное представление всех классов. Наличие изображений с несколькими повреждениями на одном автомобиле дополнительно расширяет разнообразие данных и способствует формированию устойчивых признаков у обучаемых моделей. Перед обучением была проведена валидация качества аннотаций с целью выявления и устранения возможных ошибок разметки, а также повышения согласованности классов. Примеры корректно размеченных изображений и исходных кадров с повреждениями, в том числе с множественными, представлены на рис. 1-3.

Сформированный и доработанный датасет был опубликован в открытом доступе на платформе Hugging Face [7].

III. Нейросетевые архитектуры

A. *Grounding DINO*

Grounding DINO представляет собой современную архитектуру для задач детекции объектов, ориентированную на совместную обработку визуальной и текстовой информации [8]. В отличие от классических детекторов, использующих фиксированный набор классов, Grounding DINO позволяет выполнять детекцию объектов на основе произвольных текстовый описаний, что делает модель особенно гибкой при работе с новыми или слабо формализованными категориями повреждений [9].

В основе Grounding DINO лежит архитектура, объединяющая трансформерный детектор DETR и механизмы выравнивания визуальных и языковых представлений. Модель использует двухпоточный подход, в рамках которого визуальные признаки изображения и текстовые эмбеддинги обрабатываются параллельно и взаимодействуют на уровне многоуровневых представлений [10].

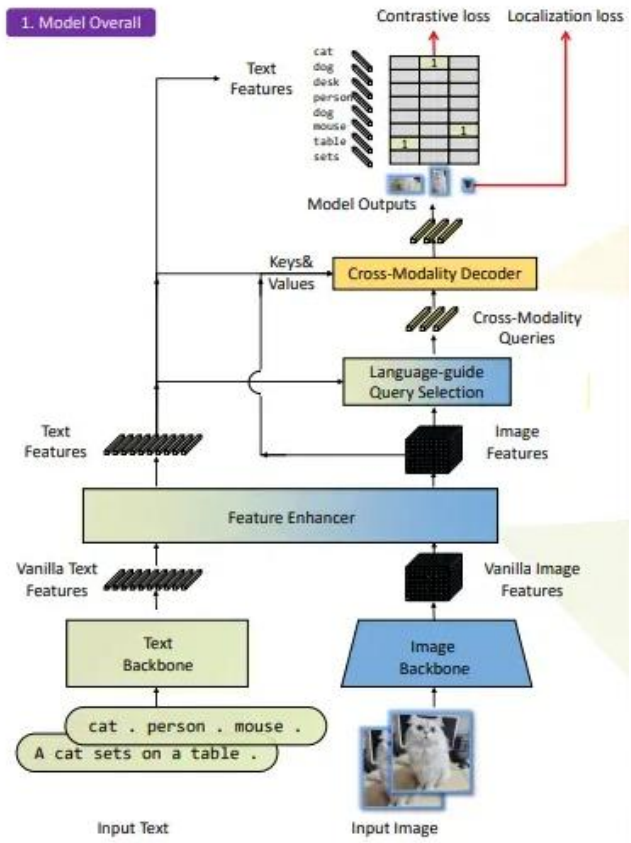


Рис. 4. Архитектура модели Grounding DINO

В представленной схеме (рис. 4) показана обобщенная архитектура Grounding DINO, иллюстрирующая принцип взаимодействия текстовых и визуальных признаков. Входное изображение проходит через визуальный бэкбон, формируя многоуровневые признаки, которые далее поступают в декодер. Параллельно текстовые описания обрабатываются текстовым энкодером, после чего их представления используются в качестве запросов для механизма кросс-модального внимания. В результате формируются выходные представления, содержащие информацию о местоположении объектов, их принадлежности к текстовым категориям и степени уверенности модели.

Важной особенностью Grounding DINO является использование контрастивного обучения, при котором модель оптимизируется таким образом, чтобы максимизировать соответствие между релевантными парами «изображение – текст» и минимизировать его для нерелевантным. Это позволяет эффективно связывать визуальные признаки с семантическими описаниями и повышает обобщающую способность модели.

Кроме того, в архитектуре применяются механизмы многоуровневой агрегации признаков, что позволяет учитывать объекты различных масштабов и форм. Такой подход особенно важен для задач анализа повреждений автомобиля, где дефекты могут существенно отличаться по размеру, форме и контрастности. Использование модели позволяет обнаруживать как крупные деформации кузова, так и мелкие локальные повреждения не требуя отдельной настройки.

B. OWLv2 (Open-World Localization v2)

OWLv2 (Open-World Localization v2) представляет собой развитие идей открытого словаря (open-vocabulary detection) и направлена на решение задачи детекции объектов в условиях отсутствия фиксированного набора классов [9]. В отличие от классических детекторов, требующих заранее определенного списка категорий, OWLv2 позволяет выполнять локализацию объектов на основе произвольных текстовых описаний, что делает модель особенно пригодной для анализа сложных и слабо структурированных визуальных сцен.

Архитектура OWLv2 основана на принципах мультимодального обучения и использует единое представление для обработки визуальных и текстовых данных. В качестве визуального энкодера применяется модифицированная версия Vision Transformer (ViT), обеспечивающая извлечение высокоуровневых признаков изображения. Текстовый поток обрабатывается отдельным языковым энкодером, формирующим векторные представления текстовых запросов, описывающих искомые объекты или их свойства.

Ключевой особенностью OWLv2 является использование совместного пространства признаков, в котором визуальные и текстовые эмбеддинги сопоставляются друг с другом. В отличие от классических детекторов, где классификация осуществляется по заранее заданным категориям, в OWLv2 сопоставление происходит на основе семантического сходства между визуальными признаками и текстовыми описаниями. Это позволяет модели эффективно обрабатывать новые или редкие категории без необходимости дополнительного обучения.

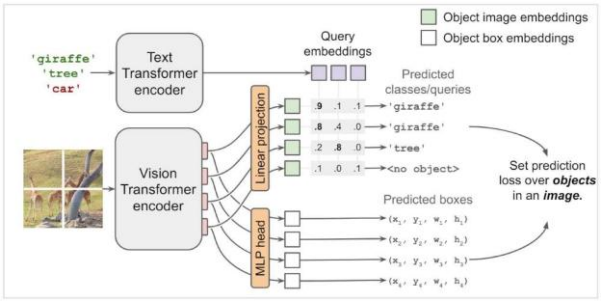


Рис. 5. Архитектура модели OWLv2

В процессе работы модель формирует набор региональных признаков, соответствующих потенциальным объектам на изображении, и сопоставляет их с текстовыми запросами с помощью механизма внимания. Такой подход позволяет одновременно решать задачи локализации и семантической интерпретации объектов. Благодаря использованию трансформерной архитектуры и глобального внимания OWLv2 демонстрирует устойчивость к вариациям масштаба, формы и освещённости объектов [10].

Дополнительным преимуществом OWLv2 является её масштабируемость. Модель поддерживает расширение словаря без повторного обучения и способна эффективно работать в условиях ограниченного количества размеченных данных. Это

делает её особенно перспективной для задач анализа повреждений автомобилей, где перечень возможных дефектов может быть широким и слабо формализованным.

IV. Сравнение

Сравним два подхода на валидационной части размеченного набора данных повреждений машин (25% от общего объёма 273 аннотации). Для оценки использовался стандартный протокол детекции + классификации, при этом:

— объект считается правильно локализованным (TP), если IoU между предсказанным прямоугольником и соответствующей разметкой ≥ 0.50;
— FP — предсказание, не соответствующее ни одной аннотации (IoU < 0.50 со всеми аннотациями);
— FN — аннотация, для которой детектор не выдал предсказание с IoU ≥ 0.50.

По введённым величинам вычислялись Precision, Recall и F1:

- $Precision = \frac{TP}{TP+FP}$ – сколько раз детектор нашёл светофор, где он действительно есть, по отношению к общему числу предсказанных светофоров;
- $Recall = \frac{TP}{TP+FN}$ – сколько светофоров 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).

ТАБЛИЦА 1. Оценка детектирующей части

Параметр	Grounding DINO	OWLv2
TP	230	250
FP	30	10
FN	43	23
Precision	88.46%	96.15%
Recall	84.25%	91.58%
F1	86.30%	93.81%

ТАБЛИЦА 2. Оценка классифицирующей части (по TP)

Модель	Число TP	Кор. класиф.	Некор. класиф.	Accuracy (на TP)
Grounding DINO	230	205	25	89.00%
OWLv2	250	235	15	94.00%

Количественные результаты оценки детектирующей части для обеих моделей представлены в таблице 1. Полученные значения показывают, что OWLv2 характеризуется большим числом истинно-положительных детекций и меньшим количеством ложных срабатываний по сравнению с Grounding DINO. Это приводит к более высоким значениям как precision, так и recall, а также к увеличению F1-меры [11, 12]. В то же время Grounding DINO демонстрирует более высокую долю пропущенных объектов, что отражается в увеличенном числе FN и снижении полноты обнаружения повреждений.

Оценка классифицирующей части проводится на множестве объектов, корректно локализованных на предыдущем этапе, то есть входящих в TP-множество детектирующей части [13]. Для каждого такого объекта проверяется соответствие предсказанного типа повреждения его истинной метке, после чего вычисляется точность классификации (accuracy). Использование данного подхода позволяет исключить влияние ошибок локализации и сосредоточиться непосредственно на качестве распознавания типов повреждений [14].

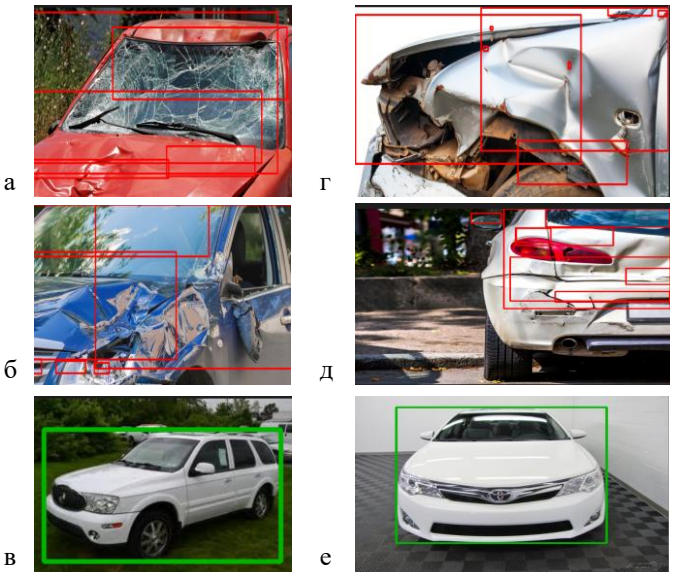


Рис. 6. Результаты тестирования обученных моделей а, б, в) Grounding DINO, г, д, е) OWLv2

Результаты классификации показывают, что OWLv2 демонстрирует более высокую точность распознавания по сравнению с Grounding DINO. Различия в accuracy могут быть связаны с тем, что более точная локализация формирует для классификатора менее зашумлённые и более информативные области изображения. В случае Grounding DINO часть ошибок классификации наблюдается для объектов с неточно определёнными границами, что затрудняет корректное выделение визуальных признаков повреждений [15].

Таким образом, сравнительный анализ показывает различия в характере ошибок двух моделей как на этапе локализации, так и на этапе классификации. Grounding DINO чаще демонстрирует пропуски объектов и ложные срабатывания, что отражается на итоговых метриках качества, тогда как OWLv2 показывает более стабильные значения показателей на обоих этапах обработки. Представленные результаты позволяют сопоставить поведение моделей в рамках единого экспериментального протокола и служат основой для дальнейшего обсуждения особенностей их применения к

задаче автоматического анализа повреждений автомобилей.

V. Заключение

В рамках данной работы были рассмотрены методы и наборы данных, используемые для обучения и тестирования нейронных сетей, предназначенных для обнаружения и классификации повреждений автомобилей [16]. Были проанализированы два подхода, основанные на моделях Grounding DINO и OWLv2, каждая из которых решает задачу детектирования, включающую локализацию объектов и определение их типа. Для обеих моделей была выполнена тонкая донастройка на собственном размеченном наборе данных, что позволило адаптировать предобученные представления к специфике решаемой задачи.

Сравнение проводилось на валидационной выборке с аннотированными изображениями повреждений. Оценивались показатели качества локализации и классификации, а полученные результаты продемонстрировали различия в характере ошибок, обусловленные особенностями архитектур и механизмов извлечения признаков. Анализ показал, что рассматриваемые модели по-разному распределяют ошибки между этапами детекции и классификации.

Следует отметить, что полученные результаты отражают эффективность конкретных реализаций моделей при фиксированных настройках обучения. Ограничивающим фактором исследования является относительно небольшой объём обучающего датасета, что может приводить к переобучению и снижению устойчивости моделей при работе с изображениями, отличающимися по условиям съёмки или типам повреждений. В дальнейшем для повышения обобщающей способности моделей целесообразно расширить и сбалансировать датасет, а также провести дополнительную оптимизацию гиперпараметров и архитектурных решений.

Литература

[1] Кожухов, А. А. Построение карты пространства вокруг автомобиля на основе видеопоследовательности / А. А. Кожухов, П. Д. Хонер // 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. 22–27. – EDN BXVECN.

[2] Леонов, И. Ю. Классификация транспортных средств компьютерным зрением / И. Ю. Леонов // 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. 46–52. – EDN JSRESQ.

[3] Ali, B., Sadekov, R. N., Tsodokova, V. V. A Review of Navigation Algorithms for Unmanned Aerial Vehicles Based on Computer Vision Systems // Gyroscopy and Navigation. – 2022. – Vol. 13. – P. 241–252. – DOI: 10.1134/S2075108722040022.

[4] Темкин, И. О. Распознавание и отслеживание дефектов дорожного полотна в реальном времени на основе комплексного использования стандартных вычислительных процедур и глубоких нейронных сетей / И. О. Темкин, М. О. Антонов // Программные продукты и системы. – 2024. – № 3. – С. 421-430. – DOI 10.15827/0236-235X.147.421-430. – EDN NZMTQA.

[5] Pazychev, D., Bakulev, K., Sadekov, R. Low-Cost Navigation System for UAV // Proceedings of the International Conference on Intelligent Navigation Systems (ICINS). – 2023. – P. 1–6. – DOI: 10.23919/ICINS51816.2023.10168469.

[6] Car Damage Dataset (CarDD) [Электронный ресурс]. – URL: https://cardd-ustc.github.io/ (дата обращения: 24.12.2025).

[7] Damaged Cars Dataset [Электронный ресурс]. – Hugging Face. – URL: https://huggingface.co/datasets/maxxq/damaged-cars-ds (дата обращения: 24.12.2025).

[8] Liu, S., Ding, W., Liu, X., Zhang, X., Wang, Y., Darrell, T. Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection // arXiv preprint arXiv:2303.05499. – 2023.

[9] Minderer, M., Girdhar, R., Joulin, A., et al. OWL-ViT: Open-Vocabulary Object Detection via Vision and Language Knowledge Distillation // European Conference on Computer Vision (ECCV). – 2022. – P. 1–14.

[10] Carion, N., Massa, F., Synnaeve, G., et al. End-to-End Object Detection with Transformers // European Conference on Computer Vision (ECCV). – 2020. – P. 213–229.

[11] Lin, T.-Y., Maire, M., Belongie, S., et al. Microsoft COCO: Common Objects in Context // European Conference on Computer Vision (ECCV). – 2014. – P. 740–755.

[12] Ren, S., He, K., Girshick, R., Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2017. – Vol. 39, No. 6. – P. 1137–1149.

[13] Redmon, J., 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.

[14] Everingham, M., Van Gool, L., Williams, C. K. I., et al. The PASCAL Visual Object Classes (VOC) Challenge // International Journal of Computer Vision. – 2010. – Vol. 88. – P. 303–338.

[15] Lin, T.-Y., Dollár, P., Girshick, R., et al. Feature Pyramid Networks for Object Detection // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2017. – P. 2117–2125.

[16] Szegedy, C., Vanhoucke, V., Ioffe, S., et al. Rethinking the Inception Architecture for Computer Vision // Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2016. – P. 2818–2826.

ГРНТИ 20.23.15
ASJC 1707
OECD 01.02.EP

Нейросетевые методы для распознавания фруктов и овощей

Н. И. Демин
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2508053@edu.misis.ru

В. С. Матвеев
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2109128@misis.ru

***Аннотация* — в работе рассматривается задача детекции и классификации фруктов на изображениях. Задача имеет практическое значение для автоматизации сортировки продуктов, контроля качества в сельском хозяйстве и розничной торговле. Для решения задачи применялась современная SOTA-модель YOLOv8s. Для обучения и тестирования был собран и размечен датасет, состоящий из изображений яблок, бананов, моркови, апельсинов, винограда, манго, огурцов, томатов и картофеля, размеченных в формате, совместимом с YOLO. Результатом работы является публичный датасет.**

Ключевые слова — компьютерное зрение, детекция светофоров, классификация фруктов, машинное обучение, автоматизация розничной торговли, YOLOv8.

I. Введение

В последние годы задачи детекции и классификации объектов на изображениях приобрели особую актуальность в связи с ростом автоматизации процессов в сельском хозяйстве, розничной торговле и производстве пищевых продуктов [1]. В этом контексте методы компьютерного зрения выступают эффективным инструментом для автоматического распознавания фруктов и овощей.

Современные алгоритмы обработки изображений и машинного обучения позволяют системам анализировать визуальные данные – изображения или видеопоток – и распознавать объекты по их характерным визуальным признакам, таким как форма, цвет и структура [2]. Особенно актуальной становится потребность в точной идентификации различных видов фруктов, что необходимо для автоматизации сортировки, контроля качества продукции и оптимизации логистики. Для этого современные системы требуют разработки высокоточных и быстрых алгоритмов, способных надежно обнаруживать и классифицировать объекты в различных условиях освещения, на фоне различных поверхностей и при разнообразных углах съемки [3].

Технологии глубокого обучения и нейросетевые модели, такие как YOLO, демонстрируют значительный прогресс в области компьютерного зрения и успешно применяются для задач детекции и классификации объектов на изображениях и видео.

Модель YOLOv8 отличается высокой скоростью обработки и эффективностью, что делает ее предпочтительным выбором для приложений с ограниченными вычислительными ресурсами и требованиями к оперативности [4]. Однако для успешного применения этих моделей в задачах распознавания требуется учет множества факторов, включая разнообразие условий освещения, фоновые объекты, а также схожие визуальные характеристики между различными видами фруктов.

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 съемки фруктов и овощей [5].

II. Наборы данных

Для решения задачи детекции и классификации фруктов был собран датасет, состоящий из изображений, взятых из открытых источников и дополнительно размеченных для обучения модели.

Датасет включает изображения девяти классов: «Apple», «Banana», «Carrot», «Orange», «Grape», «Mango», «Cucumber», «Tomato» и «Potato». Для каждого объекта было использовано по 115 изображений. Для каждого изображения имеется аннотация в виде ограничивающих прямоугольников, точно локализирующих объекты и относящих их к соответствующему классу [6]. На рисунке 1 представлен пример разметки изображений с помощью CVAT.

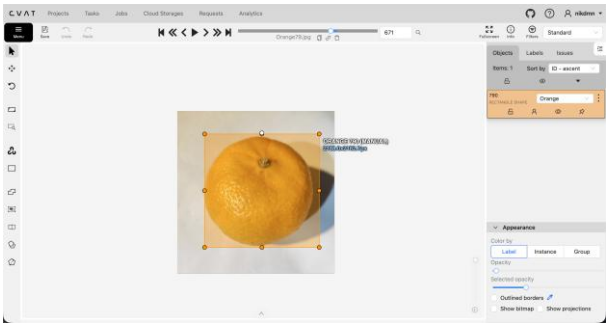


Рис. 1. Пример разметки изображений при помощи CVAT

Для оптимальной оценки качества модели данные были разделены на тренировочную, валидационную и тестовую выборки в пропорциях 70%, 20% и 10% соответственно. Такой подход соответствует современным стандартам машинного обучения и позволяет эффективно контролировать процесс обучения, а также проверять обобщающую способность модели на данных, которые раньше не видела.

Средний размер изображений составляет примерно 0.45 мегапикселя, с медианным разрешением 640×640 пикселей, что обеспечивает однородность входных данных и упрощает этапы предобработки. Размеры объектов на изображениях варьируются от небольших, находящихся на фоне, до крупных, расположенных вблизи камеры. Такое разнообразие масштабов и дистанций повышает устойчивость модели к различным условиям съемки.

Общее количество аннотаций достигает нескольких тысяч, что в среднем соответствует нескольким объектам на изображении. Высокая плотность объектов на изображениях отражает реалистичные сценарии, где фрукты могут находиться в группах или плотных скоплениях, что усложняет задачу детекции и требует от модели способности к точному разделению и локализации множества объектов в одном кадре.

III. Нейросетевые архитектуры

Для решения задачи детекции и классификации фруктов в настоящей работе была применена современная сверточная нейросетевая архитектура YOLOv8. YOLOv8 относится к семейству моделей детекции объектов в архитектуре «one-stage», что означает выполнение локализации и классификации объектов за один проход изображения через сеть. Такой подход обеспечивает высокую скорость обработки и возможность работы в реальном времени, что критично для приложений с ограниченными вычислительными ресурсами и требованиями к оперативности.

Архитектура YOLOv8 состоит из трех основных компонентов: Backbone, Neck и Head. Backbone отвечает за извлечение признаков с изображения и построен на основе блоков CSP (Cross Stage Partial) и других современных сверточных конструкций. Он позволяет сети выделять как низкоуровневые признаки, такие как границы и текстуры, так и высокоуровневые абстракции, отражающие форму и цвет объектов [7], [8].

Важной особенностью Backbone является снижение пространственного разрешения признаков с помощью операций strided convolution и max pooling, что сокращает вычислительные затраты, а блоки CSP уменьшают дублирование информации и повышают эффективность обучения.

Neck модели выполняет агрегацию признаков с разных уровней, что обеспечивает детекцию объектов разнообразного масштаба. Для этого используются комбинации FPN (Feature Pyramid Network) и PAN (Path Aggregation Network). FPN позволяет объединять признаки высокого и низкого уровня, что важно для обнаружения как мелких, так и крупных фруктов на изображениях, а PAN усиливает поток информации от низкоуровневых признаков к высокоуровневым, улучшая локализацию объектов на фоне сложных текстур. Такая конструкция обеспечивает модели возможность учитывать одновременно глобальный контекст сцены и локальные детали, что критично для точного разделения фруктов, расположенных в группах или частично перекрытых.

Head YOLOv8 формирует окончательные предсказания и выполняет три основные задачи:

- регрессия координат ограничивающих прямоугольников (bounding box regression);
- предсказание вероятности наличия объекта (objectness score);
- определение класса объектов.

Важной особенностью YOLOv8 является использование anchor-free подхода, который позволяет модели самостоятельно определять оптимальные размеры и позиции прямоугольников без необходимости заранее задавать якорные размеры, что особенно удобно при работе с объектами различного масштаба. Для повышения устойчивости модели к различным условиям съемки на этапе обучения применяется широкий спектр аугментаций, включая масштабирование, повороты, отражения и цветовые трансформации [9].

Такие методы позволяют значительно улучшить обобщающую способность модели и предотвращают переобучение на небольшом или однородном датасете. Обучение модели осуществляется с использованием комбинированной функции потерь, включающей Box Loss (IoU/CIoU) для точного позиционирования прямоугольников, Classification Loss для обучения классификации объектов и Distribution Focal Loss для более точного предсказания границ объектов. В архитектуре также применяются современные методы нормализации и активации, включая Batch Normalization и SiLU, что обеспечивает стабильное обучение и быструю сходимость сети.

Использование YOLOv8 для задачи распознавания фруктов имеет ряд преимуществ. Модель обеспечивает высокую скорость обработки, поддерживает объекты разных масштабов благодаря комбинации FPN/PAN и anchor-free head, а также демонстрирует устойчивость к разнообразным условиям съемки.

Кроме того, архитектура легко адаптируется под новые классы фруктов без необходимости значительной доработки, что делает ее идеальной для применения в автоматизации сортировки, контроля и других задач в сельском хозяйстве и пищевой промышленности.

На рисунке 2 показано схематическое изображение архитектуры YOLOv8 [10].

Для обучения использовалась предобученная на датасете COCO версия модели YOLOv8, что позволило значительно сократить время сходимости и повысить точность за счет использования уже изученных низкоуровневых признаков. Предобученные веса обеспечивают эффективное извлечение признаков с изображений и позволяют модели быстрее адаптироваться к новым классам объектов при дообучении на пользовательском датасете [10].

Обучение проводилось с использованием метода стохастического градиентного спуска с автоматическим подбором оптимизатора и начальными гиперпараметрами по умолчанию, рекомендованными для YOLOv8.

Размер изображения был установлен 640×640, что обеспечивало баланс между точностью предсказания и скоростью обработки. Размер батча выбирался исходя из объема доступной видеопамяти и составил 16

изображений на итерацию. Количество эпох обучения составляло 50, что позволяло модели стабильно сходится, избегая переобучения при достаточно разнообразном датасете. Важным этапом обучения была настройка аугментаций.

YOLOv8 использует методы случайного масштабирования, отражения, изменения яркости и контрастности, а также Mosaic и MixUp. Эти методы значительно повышают устойчивость модели к различным условиям съемки и помогают эффективно обучать сеть на ограниченном объеме данных [11].

Кроме того, для контроля процесса обучения использовались встроенные метрики качества: точность (Precision), полнота (Recall), средняя точность по порогу IoU 0.5 (mAP50) и средняя точность по диапазону порогов 0.5-0.95 (mAP50-95). Использование предобученных весов, корректная настройка гиперпараметров, эффективная аугментация данных и контроль метрик качества обеспечили высокую точность и устойчивость модели YOLOv8 при детекции и классификации фруктов, что делает ее пригодной для практического применения в задачах автоматизации сортировки и контроля качества продукции [12].

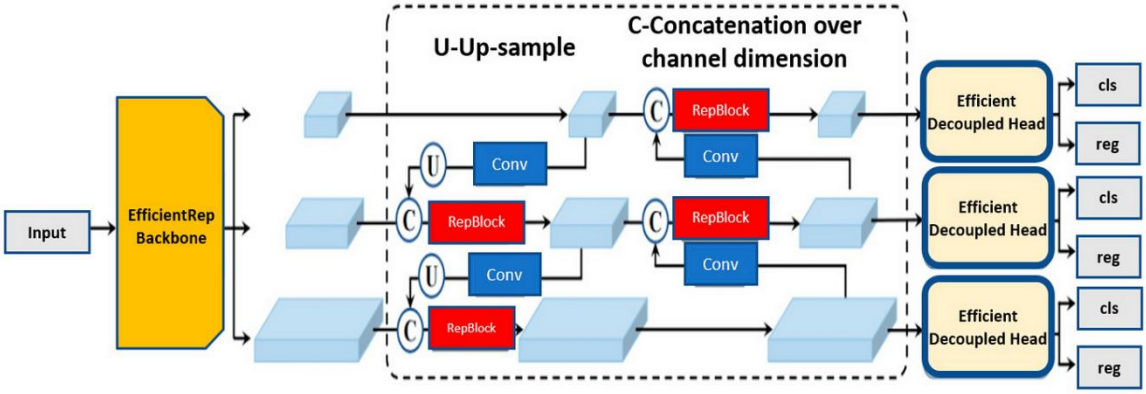


Рис. 2. Схематическое изображение архитектуры YOLOv8

IV. Результаты

Для решения задачи детекции и классификации фруктов была обучена модель YOLOv8 на датасете Fruits and Vegetables, включающем изображения девяти классов: «Apple», «Banana», «Carrot», «Orange», «Grape», «Mango», «Cucumber», «Tomato» и «Potato». Обучение проводилось в течение 50 эпох с использованием оптимизатора AdamW с начальными гиперпараметрами: learning rate (lr) – 0.01, weight decay – 5×10^{-4}.

На рисунках 3-6 изображены примеры рассматриваемых данных.

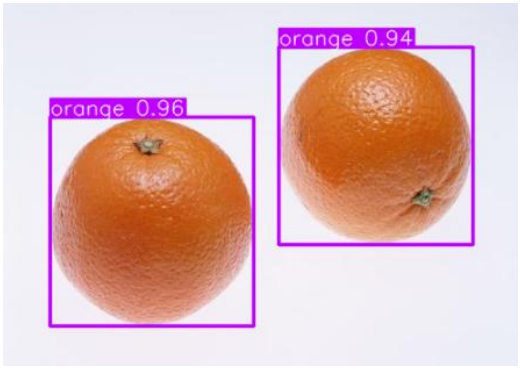


Рис. 3. Детекция класса Orange

Рис. 4. Детекция класса Apple

Рис. 5. Детекция класса Banana

Рис. 6. Детекция класса Banana

Для повышения устойчивости модели к различным условиям съемки применялись методы аугментации изображений, такие как случайные повороты, отражения, изменения яркости и контрастности, а также Mosaic и MixUp. Эти методы позволили значительно улучшить обобщающую способность модели и снизить риск переобучения на небольшом датасете [13], [14].

Функция потерь YOLOv8 является комбинированной и состоит из трех компонентов:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{box}} + \mathcal{L}_{\text{cls}} + \mathcal{L}_{\text{dfl}} \quad (1)$$

где:

- $\mathcal{L}_{\text{box}}$ – ошибка регрессии ограничивающего прямоугольника, измеряемая через IoU или его усовершенствованные версии CIoU/GIoU;
- $\mathcal{L}_{\text{cls}}$ – бинарная кросс-энтропия для классификации объектов;
- $\mathcal{L}_{\text{dfl}}$ – Distribution Focal Loss, повышающая точность предсказания границ объектов.

Функции потерь для тренировочной и валидационной выборки демонстрируют стабильное снижение по мере увеличения числа эпох (см. рис. 6 и 7). В частности, train/box_loss и val/box_loss постепенно уменьшаются с начального значения около 0.8-1.1 до примерно 0.4-0.5, что свидетельствует о корректной локализации объектов сетью. Потери классификации (cls_loss) и распределения (dfl_loss) также стабильно снижаются, показывая эффективность процесса обучения.

Для количественной оценки качества детекции используются следующие метрики:

1. Precision (точность): $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$ (2)

где TP – количество верно предсказанных объектов, а FP – количество ложноположительных объектов. Precision отражает долю корректных детекций среди всех предсказанных объектов.

2. Recall (полнота): $\text{Precision} = \frac{\text{TP}}{\text{TP+F}N}$ (3)

где FN – количество объектов, которые присутствуют на изображении, но не были обнаружены моделью. Recall показывает, какую часть всех реальных объектов модель смогла корректно идентифицировать.

3. Mean Average Precision при IoU=0.5 (mAP50): $\text{mAP50} = \frac{1}{\text{N}_\text{c}} \sum_{\text{c}=1}^{\text{N}_\text{c}} \text{AP}_\text{c}$ (4)

где N_c – количество классов, а AP_c – средняя точность для класса c, при пороге IoU 0.5. mAP объединяет информацию о точности и полноте и служит комплексной оценкой качества детекции.

На рисунках 7-9 видно, что метрики Precision и Recall быстро достигают значений выше 0.9, а mAP50 стабилизируется на уровне примерно 0.92. Это подтверждает высокую точность модели при локализации и классификации фруктов.

Графики функции потерь и метрик демонстрируют следующие закономерности:

- Функция потерь на тренировочной и валидационной выборках постепенно снижаются и не имеют резких выбросов, что указывает на корректный процесс оптимизации и отсутствие сильного переобучения;
- Метрики качества стремятся к высоким значениям уже на первых 20-25 эпохах и остаются стабильными на протяжении последующих эпох, показывая, что модель успешно обобщает на валидационной выборке;
- Незначительные колебания метрик наблюдаются лишь на отдельных эпизодах, что объясняется вариативностью объектов в батчах и случайностью аугментаций.

Экспериментальные результаты показывают, что YOLOv8 достигает высокой точности детекции фруктов при сохранении баланса между Precision и Recall. Высокие значения mAP50 подтверждают точность локализации объектов, а стабильная динамика функций потерь свидетельствует о корректной сходимости модели [15], [16].

Anchor-free подход и многоуровневая агрегация признаков через FPN/PAN позволяют модели эффективно обнаруживать объекты разного масштаба и плотности, что критично для практических сценариев, таких как автоматическая сортировка фруктов на производственных линиях или контроль качества продукции.

Таким образом, обученная модель YOLOv8 демонстрирует высокую точность и устойчивость к вариативности данных, подтверждая свою пригодность для задач автоматической детекции и классификации фруктов в реальных условиях [17].

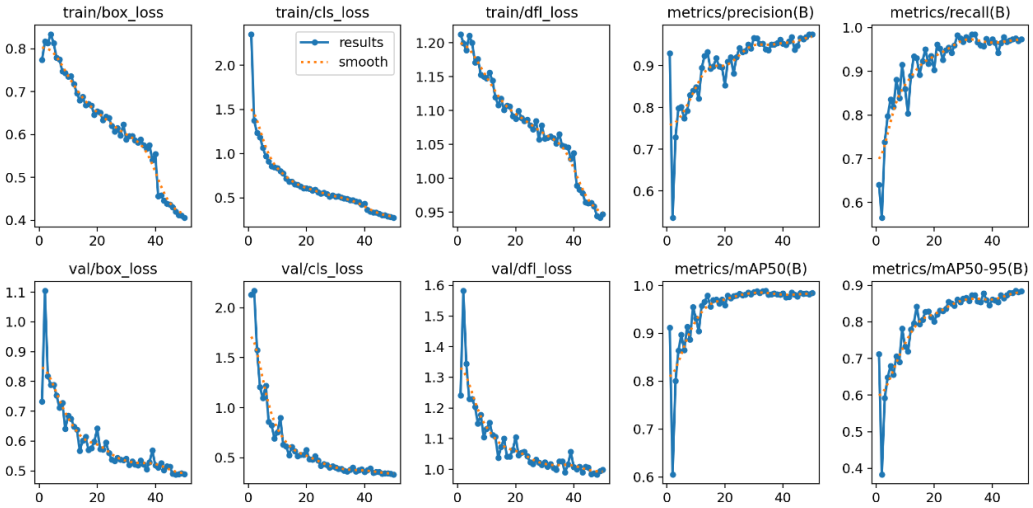


Рис. 7. Динамика функций потерь и метрик качества модели в процессе обучения

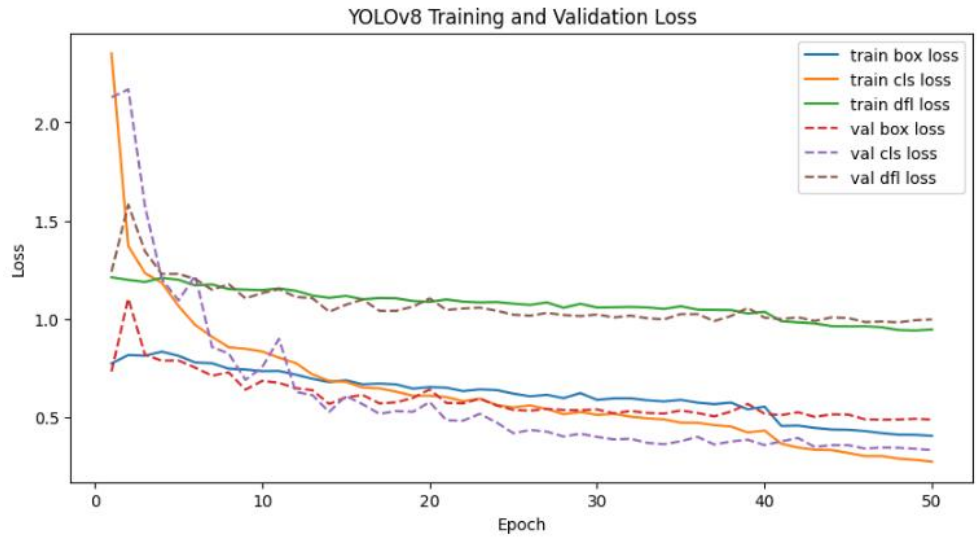


Рис. 8. Динамика функций потерь модели YOLOv8 на обучающей и валидационной выборках

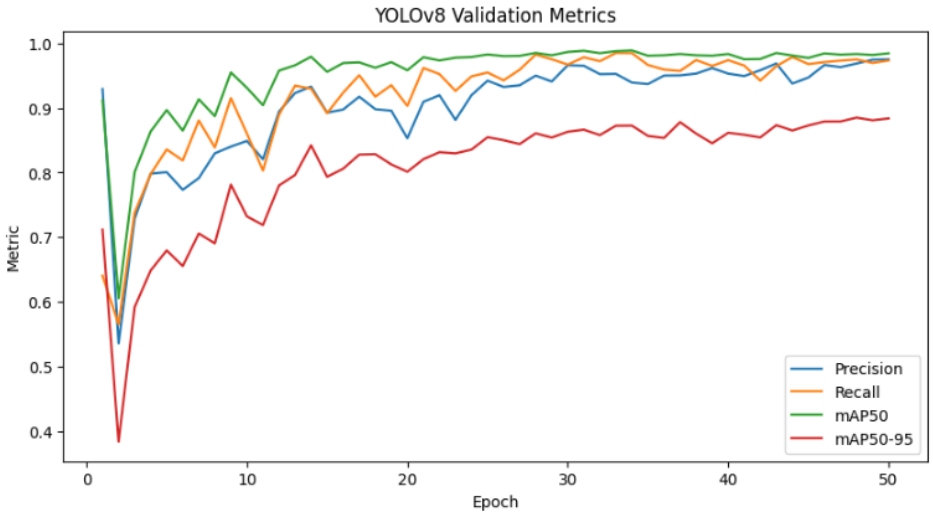


Рис. 9. Динамика валидационных метрик качества модели YOLOv8 в процессе обучения

V. Заключение

В данной работе была рассмотрена и подробно проанализирована нейросетевая архитектура YOLOv8, применяемая для задачи детекции и классификации фруктов и овощей на изображениях. Для обучения модели был использован пользовательский датасет Fruits and Vegetables, включающий 1012 изображений, отражающих разнообразные условия съемки, различные масштабы объектов и плотность расположения фруктов на изображениях.

Обучение модели проводилось с использованием предобученных весов на датасете COCO, что позволило ускорить процесс сходимости и повысить точность распознавания за счет извлечения универсальных признаков. Для оптимизации использовался стохатический градиентный спуск с адаптацией через AdamW, а также применялись методы аугментации, включая Mosaic, MixUp, случайные повороты, отражения и изменения яркости и контрастности [18].

Оценка качества модели выполнялась с использованием стандартных метрик детекции объектов: Precision, Recall и mAP50. Экспериментальные результаты показали, что модель YOLOv8 демонстрирует высокую точность и стабильность при распознавании фруктов. Графики динамики функций потерь и метрик качества на тренировочной и валидационной выборках свидетельствуют о корректной сходимости сети и отсутствии переобучения, а значения mAP50 около 0.92 подтверждают высокую эффективность локализации объектов. Модель успешно обрабатывает объекты различного масштаба и плотности, демонстрируя устойчивость к вариативности данных и сложным условиям съемки.

Таким образом, YOLOv8 является эффективным инструментом для автоматической детекции и классификации фруктов. Полученные результаты показывают, что данная архитектура подходит для практических задач автоматизации сортировки продукции, контроля качества и других применений в сельском хозяйстве и пищевой промышленности, где важна высокая точность и скорость обработки изображений [19].

Дальнейшие исследования могут быть направлены на расширение числа классов, увеличение объема и разнообразия датасета, а также экспериментальное сравнение с другими современными архитектурами детекции для улучшения обобщающей способности модели.

Литература

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

[2] R. N. Sadekov and D. B. Pazychev, “Development of a navigation system for UAV based on computer vision,” IOP Conf. Ser.: Mater. Sci. Eng., vol. 1029, no. 1, p. 012040, 2021, doi: 10.1088/1757-899X/1029/1/012040.

[3] Овчаренко, С. Д. Применение нейронных сетей в задачах классификации насекомых / С. Д. Овчаренко // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 104-109. – EDN AXVKQX.

[4] Криворот, Ю. А. Детектирование диких животных при помощи нейронных сетей / Ю. А. Криворот, Е. А. Ильяков // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 97-103. – EDN IJVNTO.

[5] И. О. Темкин, А. А. Иванова, “Интеллектуальные системы управления в агропромышленном комплексе,” Вестн. НИТУ «МИСиС», no. 5, pp. 45–52, 2023. — URL: https://elibrary.ru/item.asp?id=82295055 (дата обращения: 25.12.2025).

[6] Huggingface — Fruit-and-Vegetables-dataset —2025— Available at: https://huggingface.co/datasets/nikdmn/Fruit-and-Vegetables-dataset/tree/main (Accessed: 10.12.2025).

[7] Wang, C.-Y., and Liao, H.-Y. M. (2024) "YOLOv1 to YOLOv10: The Fastest and Most Accurate Real-Time Object Detection Systems", arXiv. Available at: https://arxiv.org/abs/2408.09332 (Accessed: December 24, 2025).

[8] Viso.ai (2025) "YOLO Explained: From v1 to v11". Available at: https://viso.ai/computer-vision/yolo-explained/ (Accessed: December 24, 2025).

[9] Rao N. (2024) ―YOLO Explained: From v1 to v11‖ // Viso.ai. Available at: https://viso.ai/computervision/yolo-explained/ (Accessed: 25 December 2025)

[10] Jocher G., Wang C., Du N. (2022) ―YOLOv8: A Trainable Ensemble of YOLO Designs‖ // Ultralytics Docs. Available at: https://github.com/ultralytics/ultralytics (Accessed: 23 December 2025)

[11] 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: 24 December 2025)

[12] 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.

[13] 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.

[14] 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.

[15] 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.

[16] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards realtime 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] 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.

[19] 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

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

Распознавание дефектов картофеля с помощью свёрточной нейронной сети

Ф. Д. Егоров
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2514205@edu.misis.ru

***Аннотация* — В сельском хозяйстве на этапах сбора картофеля и его извлечения из хранилищ производится сортировка на предмет дефектов и болезней, что требует дополнительного времени и рабочей силы, и в то же время может быть автоматизировано с помощью машинного зрения. В работе сравнивается производительность архитектур свёрточных нейронных сетей MobileNetV3 и ResNet в задаче классификации изображений картофеля по признаку наличия дефектов в реальном времени, произведена корректировка процесса обучения с учётом наличия неточности известного характера в разметке набора данных.**

Ключевые слова — автоматизация, классификация изображений, машинное зрение, свёрточные нейронные сети.

I. Введение

На различных этапах обработки картофеля, начиная с выращивания и заканчивая отправкой на предприятия потребления или реализации, клубни могут получать различные виды дефектов. Некоторыми из таковых являются дефекты формы, механические повреждения, укусы вредителей, плесень и зелёный бок. Для экономии места в хранилищах и исключения попадания дефектного картофеля к потребителю при его перемещении на склад и из склада производится сортировка, для которой используются конвейерные линии с ручными инспекционными столами. Это порождает потребность в дополнительной рабочей силе, которая имеет сезонный характер. В то же время, процесс сортировки основан почти исключительно на визуальной оценке клубня и не предполагает как сложной процедуры анализа, так и сложных механических операций. В связи с этим в настоящее время наблюдается повышенный интерес к автоматизации сельскохозяйственных процессов, в том числе с помощью машинного зрения.

Одним из широко используемых в промышленных приложениях методов машинного зрения являются свёрточные нейронные сети (Convolutional neural networks, CNN). Этот метод является продолжением концепции глубокого обучения и основан на принципах работы зрительной коры мозга живых организмов, в которой каждый нейрон связан только с соседними, что позволяет группировать стимулы от геометрически близких точек. CNN также основываются на свойстве инвариантности содержания изображения относительно параллельного переноса — иными словами, все стороны и регионы изображения считаются равноправными и существенным является только относительное расположение объектов.

Алгоритмы на основе свёрточных нейронных сетей применяются в таких сферах, как оценка среды при управлении беспилотным общественным транспортом [1] и аэронавигация беспилотных летательных аппаратов [2]. Они показывают высокую точность как в задачах распознавания и классификации большого количества существенно различающихся объектов [3], так и в задачах классификации визуально похожих объектов с тонкими различиями [4]. Также основанные на CNN архитектуры способны проводить сегментацию изображений [5] и определять конфигурацию составных объектов с множеством подвижных частей, например, тела человека [6].

Целью работы является сравнение архитектур свёрточных нейронных сетей, показавших высокую производительность в других задачах классификации в реальном времени, в контексте задачи классификации единичных клубней картофеля по признаку наличия дефектов. Предполагается, что такая сеть будет работать в тандеме с нейросетью-детектором, предсказывающей описывающие прямоугольники для множества клубней на цельном снимке, с помощью которых путем вырезания будет сгенерирован вход модели на следующем шагу.

II. НАБОР ДАННЫХ

Для сбора данных были использованы наборы картофеля, в каждом из которых клубни либо не имели дефекта, либо имели один и тот же известный тип дефекта из следующих:

- дефект формы;
- повреждения вредителями;
- зелёный бок.

Съемка картофеля осуществлялась при его перемещении по роликовому конвейеру, при этом камера располагалась сверху с расположением оптической оси перпендикулярно плоскости движения клубней. При движении по конвейеру такого типа объекты вращаются по оси, поперечной направлению движения, что позволяет получить кадры с разных углов. В результате для каждого набора картофеля был получен отдельный набор кадров, на каждом из которых изображены клубни с одним типом дефекта, либо без дефектов вообще.

Рис. 1. Изображение из набора кадров до разметки и обработки.

Стоит отметить, что дефекты клубня могут быть не видны с определённого угла. В связи с этим, во всех наборах, кроме здорового картофеля, присутствует доля кадров, по которым заведомо нельзя определить дефект ввиду его расположения на противоположной стороне клубня. Иными словами, датасет содержит заметную долю ложноотрицательных примеров, которые необходимо учитывать при обучении. Эта особенность датасета связана с существенным повышением трудозатрат на разметку при анализе каждого изображения по отдельности, необходимости привлечения экспертов, и повышенной доле ошибок разметки в связи со спецификой задачи.

Для приведения датасета к виду, доступному для подачи на вход моделей, отдельные объекты были вырезаны из изображений, масштабированы с сохранением соотношения сторон и расположены поверх квадратного монотонного изображения 256x256 пикселей нейтрального серого цвета (RGB 127, 127, 127) с выравниванием по центру. Коэффициент масштабирования был выбран как максимальный, при котором фрагменты изображений с высотой или шириной, превышающей среднее значение ровно на 2 стандартных отклонения были полностью вписаны в квадрат. Набор данных был разбит на две части: тренировочную и тестовую. Тестовая часть составляла 10% от общего числа изображений и имела такие же пропорции классов, а также была отобрана вручную для исключения упомянутых ранее ложноотрицательных примеров. Подготовленный набор данных размещён в открытом доступе на платформе Hugging Face [8].

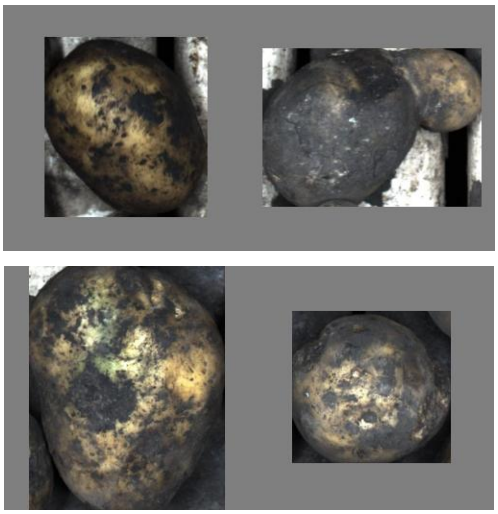


Рис. 2. Примеры из подготовленного набора данных.

Таблица 1

Количество экземпляров классов в тестовой и тренировочной выборках

Название класса (метка)	**Количество экземпляров**	
	Трен.	**Тест.**
Без дефекта (0)	471	53
Зелёный бок (1)	330	37
Повреждение вредителями (2)	241	27
Дефект формы (3)	110	16

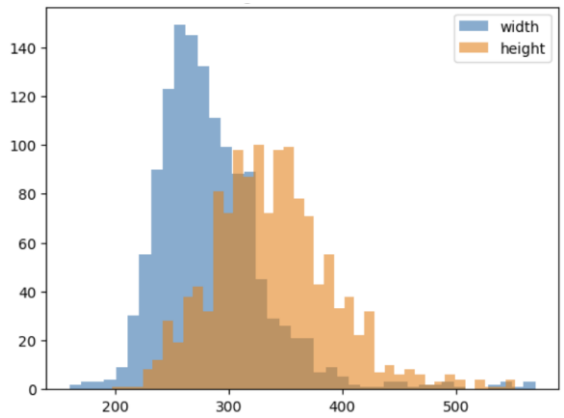


Рис. 3. Гистограмма распределения ширины (width) и высоты (height) описывающих прямоугольников объектов в наборе данных.

III. Используемые модели

Для поиска оптимального решения были обучены модели ResNet18, ResNet34, MobileNetV3_small и MobileNetV3_large. Модели были инициализированы весами, предобученными на датасете ImageNet1K.

Архитектуры семейства ResNet содержат блоки с параллельно идущими обходными связями, что позволяет информации “обходить” слои и затем поэлементно складываться с выходом блока. Такая архитектурная особенность позволяет смягчить проблему *угасающих градиентов*: при обучении модели с большим количеством последовательных слоёв градиент ошибки затухает при его обратном распространении, в результате чего параметры начальных слоёв практически не изменяются. Это явление накладывает практическое ограничение на выбор числа слоёв в архитектуре, что затрудняет построение более сложных моделей. В свою очередь, связи с пропуском позволяют градиенту распространиться на верхние слои с существенно меньшим затуханием, что позволило использовать в ResNet большее число параметров.

Модель MobileNet была разработана с целью получения наилучшего соотношения точности классификации к вычислительным затратам на запуск модели и показала высокие результаты в промышленных приложениях [7], поэтому она может быть рассмотрена для использования в автоматических сортировочных комплексах с ограниченной вычислительной мощностью.

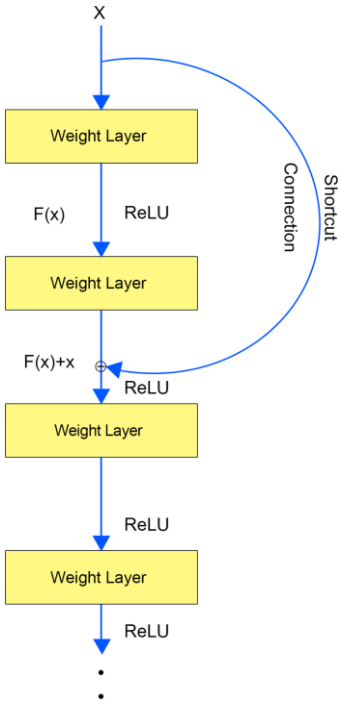


Рис. 4. Схематическое изображение связи с пропуском слоёв, используемой в ResNet.

IV. Обучение

Обучение нейронных сетей производилось с использованием модуля PyTorch языка Python. Для расширения тренировочной выборки набора использовались аугментации — случайные преобразования определённого типа, применение которых гарантированно сохраняет содержательность данных в данной задаче. В работе были использованы следующие аугментации, взятые из модуля imgaug:

- горизонтальная и вертикальная симметрия;
- поворот на угол, кратный *90°*;
- поворот на случайный угол от *-15°* до *15°*;

- относительное изменение яркости от 0,8 до 1,2;
- относительное изменение контраста от 0.8 до 1,2.
- аддитивный гауссовский шум.

Каждое из вышеперечисленных преобразований выполнялось независимо с вероятностью 0,5. Использование аугментаций позволило дополнить набор данных, что помогает улучшить обобщающую способность модели.

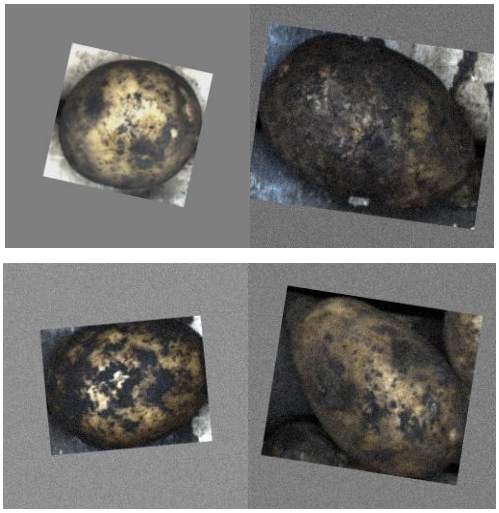


Рис. 5. Аугментированные изображения.

В основном цикле обучения для всех четырёх моделей использовались одинаковые параметры, указанные в Таблице 2. Валидационная выборка выбиралась из тестовой случайно перед обучением и составляла 0,1 от общего числа примеров.

Таблица 2

Параметры обучения

Параметр	**Значение**
Оптимизатор	ADAM
Learning rate	10^{-4}
Количество эпох	50
Размер батча	32

Для введения поправки на присутствие в наборе данных ложноотрицательных примеров, при обучении в качестве функции потерь была использована модифицированная кросс-энтропия. Доля ошибок в части датасета с дефектами была оценена как $\alpha = 0{,}3$. Для вычисления ошибки предсказания на примерах, имеющих метку “Без дефекта” с индексом, применялась обыкновенная кросс-энтропия. Для остальных меток классов ошибка вычислялась как среднее взвешенное кросс-энтропии относительно метки самого класса и кросс-энтропии относительно отрицательной метки.

Для каждой архитектуры динамика изменения функции потерь на тренировочной и валидационной выборках в зависимости от эпохи представлена на Рис. 6-9.

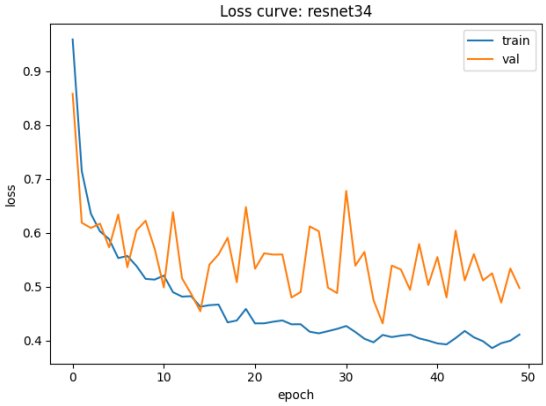


Рис. 6. График функции потерь для ResNet34.

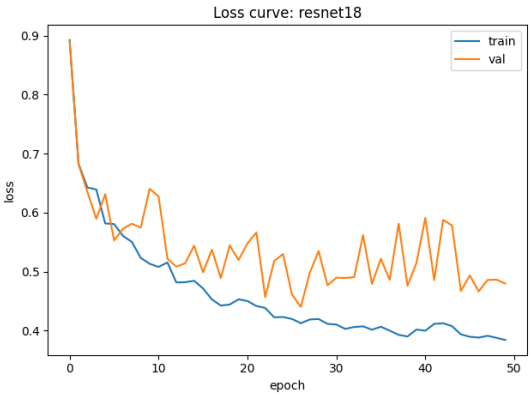


Рис. 7. График функции потерь для ResNet18.

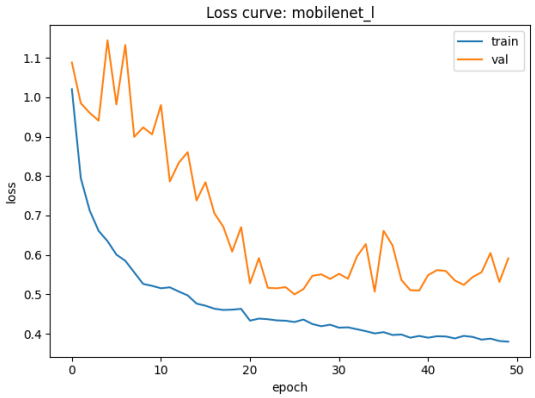


Рис. 8. График функции потерь для MobileNet_large.

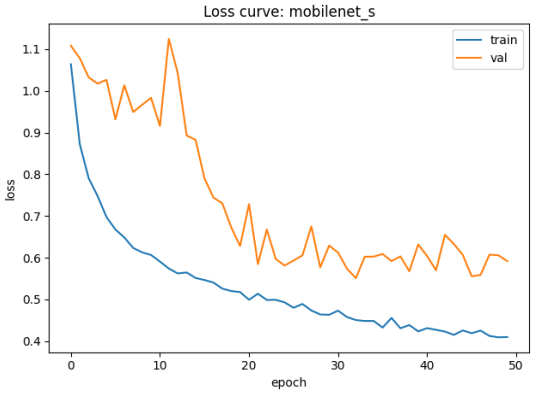


Рис. 9. График функции потерь для MobileNet_small.

V. ТЕСТИРОВАНИЕ

Качество предсказания моделей было оценено с помощью двух метрик: Accuracy (точности), определяемой как доле правильно классифицированных объектов во всей выборке и F1-Score, определяемому как среднее гармоническое Precision (избирательности) и Recall (полноты). Для получения метрики F1 задача была рассмотрена как бинарная классификация без необходимости различения отдельных дефектов, то есть классы 1-3 считались эквивалентными и принимались как обобщенный положительный класс, так как в рассматриваемой задаче отделение здорового картофеля от дефектного имеет большую значимость, чем различение разных типов дефектов между собой. В результате тестирования были получены значения, приведенные в Таблице 3.

Таблица 3

Производительность обученных моделей

Архитектура	**Accuracy**	**F1-Score**
ResNet34	**0,970**	**0,991**
ResNet18	0,940	0,975
MobileNetV3_l	0,940	0,968
MobileNetV3_s	0,911	0,947

Для исследования применимости моделей в условиях реального времени также было проведено измерение времени на один запуск каждой модели с использованием 8 ядер процессора AMD Ryzen AI 9 HX 370. Результаты измерений представлены в Таблице 4.

Таблица 4

Скорость работы используемых моделей

Архитектура	**Время, мс.**	**FPS**
ResNet34	21,016	47,6
ResNet18	11,198	89,3
MobileNetV3_l	8,529	117,2
MobileNetV3_s	**4,439**	**225,3**

Дополнительно был проведен анализ ошибок, допущенных сетью на тестовой выборке. В результате были обнаружены следующие факты:

- Наиболее частые ошибки возникают при определении невыраженных дефектов формы и повреждений вредителями, представленных небольшими углублениями.
- Модели семейства MobileNet корректно классифицируют все изображения без дефектов, но при этом чаще ошибаются на примерах с положительными метками, уступая в целом в точности ResNet.
- Зелёный бок обнаруживается и отличается от других дефектов лучше всего.

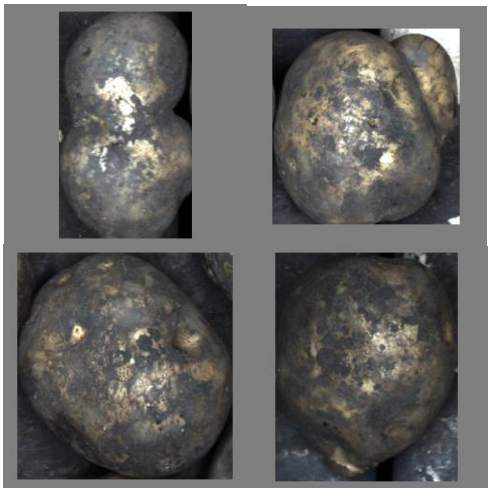


Рис. 10. Изображения, при классификации которых нейронные сети ошибаются чаще всего.

ЗАКЛЮЧЕНИЕ

Рассмотренные архитектуры показали достаточно высокую точность на используемом наборе данных и высокую скорость обработки, достигнув значения F1-метрики, равного 0,991 для отличения дефектного картофеля от здорового и затрачивая при этом на обработку одного изображения не более 22 мс. Это позволяет сделать вывод о целесообразности применения исследованных архитектур для анализа видеопотока в системах автоматической сортировки картофеля. Тем не менее, максимальный достигнутый показатель точности для различения всех четырёх классов оказался равен 0,970, что говорит о наличии существенного числа ошибок. Таким образом, результаты могут быть улучшены с помощью расширения набора данных, исследования других архитектур, а также устранения имеющихся ошибок разметки, связанных с методом сбора набора данных.

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] Ю. А. Криворот, Е. А. Ильяков, Детектирование диких животных при помощи нейронных сетей // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – 142 с. 97-103

[4] И. Б. Алексеев, П. Е. Злакоманов, ИИ в детекции фейков: Анализ подлинности лиц // 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. – 142 с. 97-103

[5] С. Д. Киселёв, А. В. Алтунян, Глубокие нейросетевые подходы к сегментации нефтяных разливов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30–31 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – 113 с. 8-12.

[6] 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.

[7] Howard A. G., Zhu M., Chen B., et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications // arXiv preprint arXiv:1704.04861, 2017.

[8] Huggingface — Potato defects. — 2025 — Available at: https://huggingface.co/datasets/servoskinner/potato-defects (Accessed: 25.12.2025).

ГРНТИ 28.00.00, 28.23.00, 28.23.15
ASJC 1702, 1707, 1712
OECD 1.2, 1.2.2, 1.2.3

Анализ эффективности методов компьютерного зрения для видеоаналитики посетителей в розничной торговле

И. Д. Ермоленко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2508898@edu.misis.ru

Аннотация — рассматривается задача видеоаналитики посетителей розничного магазина на основе методов компьютерного зрения и многообъектного трекинга в условиях перекрытий, плотного потока и переменного освещения. Для решения задачи исследованы два подхода: детектор YOLO в сочетании с трекером ByteTrack и детектор RT-DETR в сочетании с трекерами DeepSORT/StrongSORT; сравнение выполнено по устойчивости идентификаторов, фрагментации траекторий и вычислительной сложности. Показано, что YOLO + ByteTrack обеспечивает более высокую скорость и меньшие вычислительные затраты, тогда как RT-DETR + DeepSORT/StrongSORT лучше сохраняет идентичность объектов в сложных сценах; результаты могут быть использованы при выборе архитектуры для практических систем видеоаналитики.

Ключевые слова — видеоаналитика, детектирование объектов, компьютерное зрение, многообъектный трекинг, розничная торговля, трансформерные детекторы, трекинг по детекциям.

I. Введение

В настоящее время системы видеоаналитики на основе методов компьютерного зрения активно внедряются в розничную торговлю и используются крупнейшими мировыми и отечественными компаниями для решения широкого круга задач. Компании Amazon, Walmart, Alibaba, X5 Retail Group, «Магнит» и другие применяют интеллектуальные системы наблюдения для подсчёта посетителей, анализа их перемещений, распределения клиентских потоков, оптимизации работы персонала и повышения эффективности использования торговых площадей[1]. Такие системы позволяют получать объективные данные о поведении покупателей без вмешательства человека и становятся неотъемлемым элементом современных цифровых магазинов[2].

Ключевым элементом подобных решений является надёжное обнаружение и отслеживание людей в видеопотоке с камер наблюдения. Задача усложняется рядом факторов, характерных для реальной торговой среды: переменным освещением, частыми перекрытиями людей друг другом и объектами интерьера (стеллажами, колоннами), плотными потоками покупателей, вариативностью углов съёмки и перспективы[3]. Для её решения применяются технологии компьютерного зрения и методы глубокого обучения, показавшие высокую эффективность в задачах обнаружения и трекинга объектов на видео[4].

В литературе описаны различные подходы к детектированию и отслеживанию людей в видеопотоке. Современные детекторы объектов общего назначения, такие как архитектуры семейства YOLO и модели на базе трансформеров, зарекомендовали себя в задачах видеонаблюдения и анализа сцен с большим количеством объектов[5]. В то же время для практического применения важно не только качественное обнаружение объектов, но и устойчивое сохранение идентичности каждого человека при его перемещении, временных исчезновениях из поля зрения камеры и перекрытиях[6]. Для этого используются алгоритмы многообъектного трекинга, среди которых широко применяются ByteTrack, DeepSORT и StrongSORT[7].

Методы глубокого обучения требуют значительных объёмов данных и вычислительных ресурсов, однако именно они обеспечивают наилучшие результаты в реальных условиях эксплуатации видеоаналитических систем. В данной работе рассматриваются и сравниваются два современных подхода к видеоаналитике посетителей в розничной торговле: комбинация детектора YOLO с трекером ByteTrack и комбинация RT-DETR с трекерами DeepSORT/StrongSORT[8]. Сравнение проводится на видеоматериалах, полученных в реальной торговой точке, с целью анализа точности детекции, качества трекинга и устойчивости алгоритмов в условиях перекрытий и сложной динамики сцены.

II. Наборы данных

В данной работе для проведения экспериментов и оценки эффективности методов видеоаналитики посетителей розничного магазина использовались два типа наборов данных: локальный видеодатасет, собранный в реальной торговой точке, а также открытые наборы данных, на которых были предварительно обучены применяемые нейросетевые модели.

A. Локальный видеодатасет торговой точки

Основным источником данных в рамках работы является локальный видеодатасет, полученный с камеры видеонаблюдения магазина одежды. Видеозаписи были сняты с фиксированной точки обзора и отражают реальные условия функционирования торгового зала: присутствие покупателей, манекенов, кассовой зоны, входной группы, а также характерные для розничной торговли ситуации перекрытия объектов, изменения плотности потока посетителей и неравномерного освещения[9].

Разрешение видеопоследовательностей составляет 1920×1080 пикселей, частота кадров соответствует стандартным параметрам систем видеонаблюдения. Разметка данных вручную не проводилась, так как цель работы заключается не в обучении моделей, а в анализе и сравнении готовых архитектур детекции и многообъектного трекинга в реальных условиях эксплуатации. Данный видеодатасет использовался для тестирования алгоритмов обнаружения людей, отслеживания их траекторий, а также расчёта статистических характеристик поведения посетителей, включая время пребывания в торговом зале и распределение по функциональным зонам[10].

На рисунках 1 - 2 представлены кадры с обзорных камер розничного магазина.

Рис. 1. Пример кадра из записи левой обзорной камеры

Рис. 2. Пример кадра из записи правой обзорной камеры

На рисунке 3 представлен кадр с камеры возле кассы.

Рис. 3. Пример кадра из записи камеры возле кассы

На рисунке 4 представлен кадр с камеры возле входа в магазин.

Рис. 4. Пример кадра из записи камеры возле входа в магазин

B. Microsoft COCO (Common Objects in Context)

COCO является одним из наиболее распространённых эталонных наборов данных для задач детекции объектов[11]. Он содержит порядка 330 тыс. изображений, из которых более 200 тыс. имеют разметку, и включает 80 категорий объектов (в том числе класс person), а также аннотации в виде ограничивающих прямоугольников (bounding boxes) и других типов разметки.

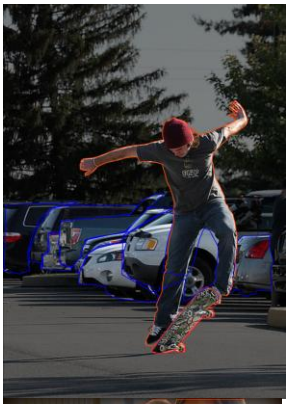
Модели семейства Ultralytics YOLO (включая современные версии) типично распространяются в виде весов, обученных на COCO с 80 классами, что обеспечивает готовность детектора к обнаружению людей без дополнительного обучения под конкретный магазин.

Для RT-DETR также доступны предобученные варианты (например, RT-DETR-L/RT-DETR-X), качество которых обычно оценивается на COCO val2017 (AP) — то есть модель изначально оптимизирована под стандартную задачу детекции объектов на “универсальных” сценах.

На рисунке 5 представлены примеры изображений, которые использовались для обучения, валидации и тестирования модели Microsoft COCO.

а

б



в

г

Рис. 5. Примеры изображений для обучения модели

III. Нейросетевые архитектуры

A. Детектор YOLO в сочетании с трекером ByteTrack

Архитектуры семейства YOLO (You Only Look Once) относятся к классу однопроходных детекторов объектов, в которых задача локализации и классификации объектов формализуется как отображение входного изображения в множество предсказанных ограничивающих прямоугольников и соответствующих им классов в рамках одного прямого прохода нейронной сети[12]. Пусть входное изображение представляется в виде тензора, где H и W — пространственные размеры изображения, C — число цветовых каналов: $I \in R^{H \times W \times C}$.

Детектор YOLO реализует функцию, где b_i — параметры ограничивающего прямоугольника, c_i — класс объекта, s_i — уверенность предсказания, а θ — параметры нейросети: $f_\theta(I) = \{(b_i, c_i, s_i)\}^N_{i=1}$. Такой подход обеспечивает высокую скорость работы за счёт отсутствия этапов генерации регионов интереса и делает YOLO популярным решением для задач реального времени, включая видеонаблюдение и видеоаналитику.

Современные версии YOLO используют глубокие сверточные нейронные сети с многоуровневым извлечением признаков. Формирование карт признаков осуществляется на нескольких уровнях абстракции, что позволяет описывать как локальные, так и более глобальные характеристики объектов. Многошкальная детекция реализуется путём предсказания ограничивающих прямоугольников на различных разрешениях карт признаков, что особенно важно при обнаружении людей, находящихся на разных расстояниях от камеры. Для оценки качества локализации используется метрика Intersection over Union (IoU), определяемая как

$$IoU = \frac{B_{pred} \cap B_{gt}}{B_{pred} \cup B_{gt}} \quad (1),$$

где B_{pred} и B_{gt} — предсказанный и истинный ограничивающие прямоугольники соответственно. Предобучение моделей на крупном открытом наборе данных COCO позволяет минимизировать ошибку обобщения и обеспечивает высокую устойчивость детектора при работе с видеоданными из реальных сцен[13].

Для обеспечения устойчивого отслеживания людей во времени в данной работе детектор YOLO используется в сочетании с алгоритмом многообъектного трекинга ByteTrack. Алгоритм ByteTrack относится к классу методов tracking-by-detection, в которых задача трекинга формулируется как задача ассоциации детекций между последовательными кадрами. Пусть D_t — множество детекций на кадре t, а T_{t-1} — множество активных треков на предыдущем кадре. Ассоциация осуществляется путём минимизации функции стоимости

$$C_{ij} = 1 - IoU\left(b_i^t, b_j^{t-1}\right) \quad (2),$$

где b_i^t и b_i^{t-1} — ограничивающие прямоугольники детекции и трека. Особенностью ByteTrack является учёт детекций с низкой уверенностью, которые обычно отбрасываются на этапе детекции, что позволяет уменьшить количество разрывов траекторий и повысить устойчивость трекинга в условиях частичных перекрытий и временных пропусков обнаружения объектов[14].

Преимуществом комбинации YOLO и ByteTrack является высокая вычислительная эффективность, обусловленная линейной сложностью ассоциации и отсутствием дополнительных нейросетевых моделей для извлечения признаков внешнего вида объектов. Это обеспечивает стабильную работу системы при плотном потоке посетителей и делает данный подход практичным для применения в системах видеоаналитики розничной торговли, где требуется баланс между точностью, скоростью обработки и вычислительными затратами.

На рисунке 6 изображена схема ByteTrack, где используется детекция и двухступенчатая ассоциация с учётом низко-уверенных боксов, что является ключевым отличием от базового SORT/SORT-like[15].

На рисунке 7 представлен пример работы системы, где ведется подсчет количества посетителей с присвоением каждому уникального ID, а также зонирование магазина на три области: зал, вход и касса.

На рисунке 8 представлен пример обработки частного случая распознавания посетителей в виде манекена. Если распознанный объект в течение некоторого времени не перемещается в пространстве, он помечается как «MANNEQUIN» и не идет в статистику по подсчету количества посетителей и времени их пребывания как в магазине, так и в конкретной его области.

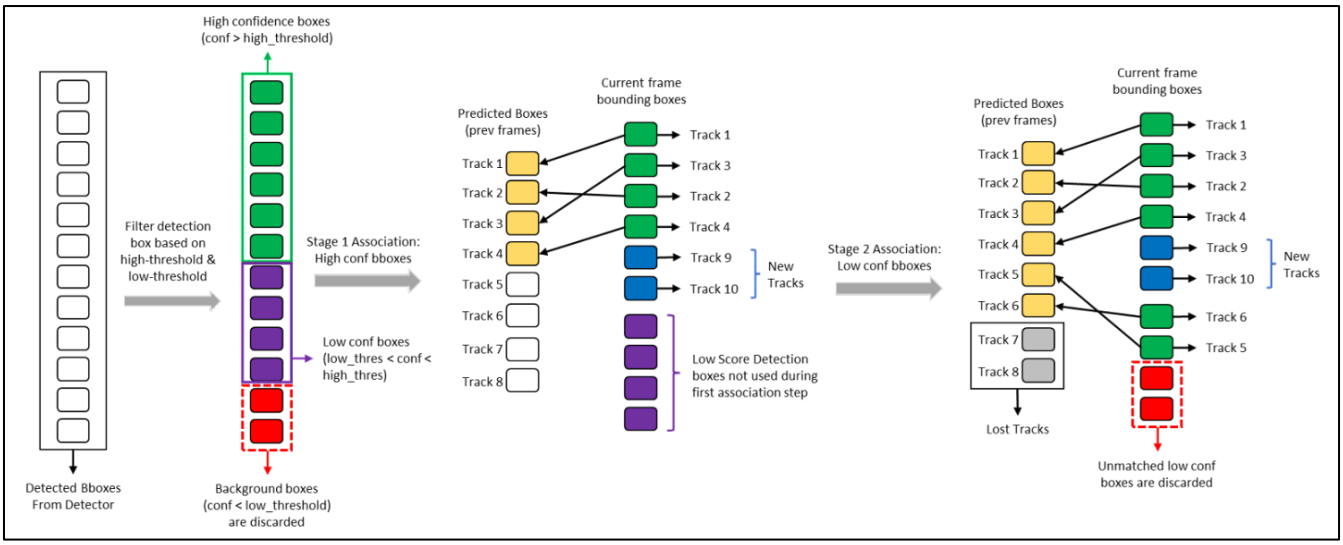


Рис. 6. Архитектура алгоритма ByteTrack: первичный и вторичный этапы ассоциации детекций между кадрами

Рис. 7. Пример работы системы с подсчетом количества посетителей и зонированием магазина

Рис. 8. Пример обработки частного случая распознавания посетителей в виде манекена у витрины

B. RT-DETR и алгоритмы многообъектного трекинга

Второй рассматриваемый подход основан на использовании архитектуры RT-DETR (Real-Time Detection Transformer), относящейся к классу детекторов объектов на базе трансформеров. В отличие от сверточных детекторов, в которых обработка изображения осуществляется преимущественно локальными фильтрами, RT-DETR использует механизм самовнимания (self-attention), позволяющий учитывать взаимосвязи между всеми пространственными областями сцены. За счёт этого модель способна анализировать изображение в глобальном контексте, что особенно важно при наличии большого количества объектов, перекрытий и сложных пространственных конфигураций, характерных для видеосъёмки в торговых залах[16].

Архитектура RT-DETR представляет собой модификацию классической модели DETR, оптимизированную для работы в условиях, близких к реальному времени. В основе модели лежит трансформерная схема с энкодером и декодером, где энкодер формирует обобщённое представление сцены на основе извлечённых признаков, а декодер оперирует фиксированным числом объектных запросов, каждый из которых потенциально соответствует одному объекту на изображении[17]. Такой подход позволяет отказаться от использования якорных прямоугольников и процедур подавления немаксимумов, характерных для сверточных детекторов, что упрощает архитектуру и снижает количество избыточных предсказаний.

Оптимизация RT-DETR основана на глобальном сопоставлении предсказанных объектов с эталонной разметкой, что обеспечивает однозначное соответствие между объектами сцены и выходными предсказаниями модели. Благодаря этому достигается более стабильная локализация объектов и уменьшается вероятность дублирования детекций. Предобучение RT-DETR на крупном наборе данных COCO позволяет модели эффективно обнаруживать людей в разнообразных условиях съёмки, включая различные масштабы объектов, уровни освещённости и плотность потока людей в кадре.

Для решения задачи многообъектного трекинга RT-DETR используется в сочетании с алгоритмами DeepSORT или StrongSORT, которые относятся к классу методов tracking-by-detection. Эти алгоритмы расширяют классический подход SORT за счёт использования дополнительных признаков внешнего вида объектов, извлекаемых с помощью специализированных сверточных нейронных сетей. Такие признаки позволяют учитывать визуальное сходство между объектами и повышают

устойчивость ассоциации идентификаторов при длительных перекрытиях, пересечениях траекторий и временных исчезновениях объектов из поля зрения камеры[18].

На рисунке 9 показана общая структура детектора RT-DETR. Входное изображение поступает в сверточный backbone, где формируются многоуровневые карты признаков, отражающие как локальные детали, так и более глобальные особенности сцены. Далее извлечённые признаки передаются в гибридный (эффективный) энкодер, который агрегирует информацию по всему изображению и учитывает взаимосвязи между различными областями сцены. На следующем этапе декодер трансформера обрабатывает фиксированный набор объектных запросов (object queries) и формирует итоговые предсказания, включающие координаты ограничивающих прямоугольников и вероятности принадлежности объектов к классам. Такая архитектура позволяет учитывать глобальный контекст сцены, снижать количество дублирующих детекций и обеспечивает устойчивую работу детектора в сложных сценах при сохранении приемлемой скорости обработки.

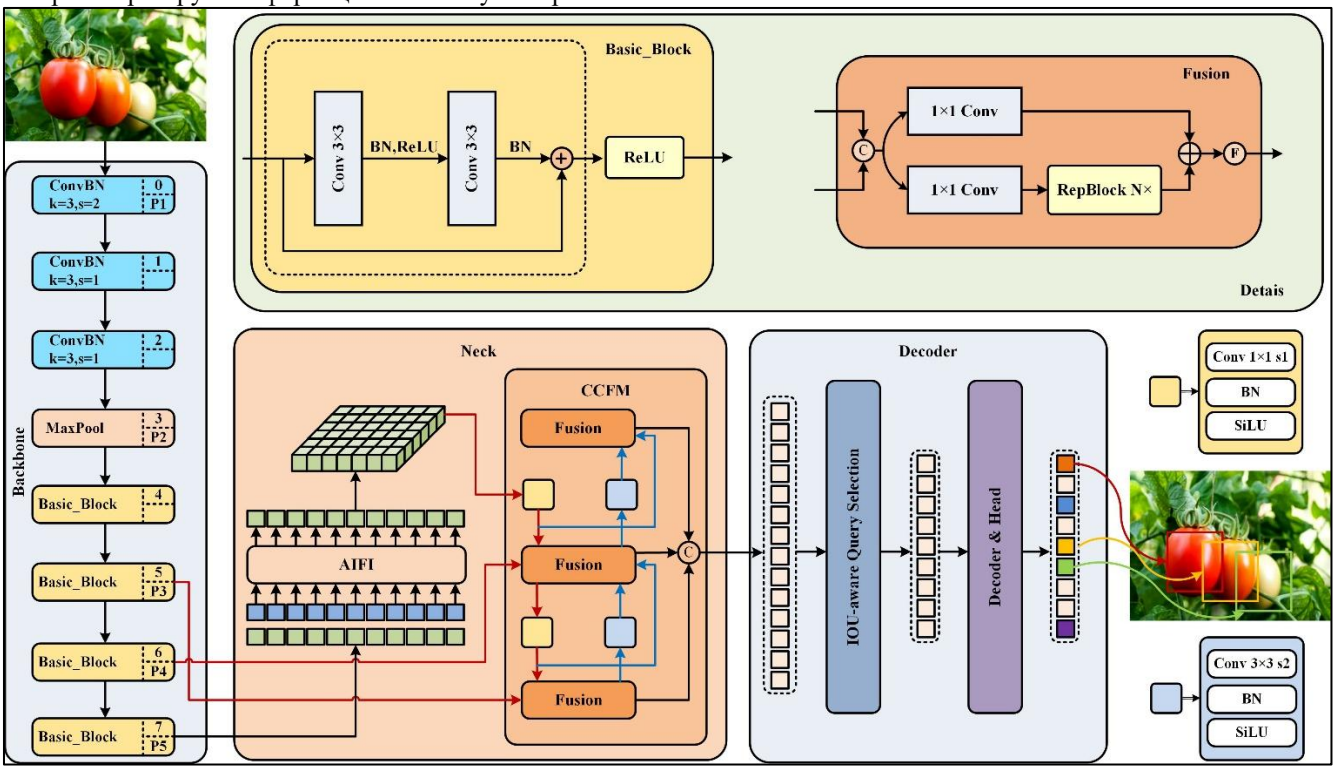


Рис. 9. Общая структура детектора RT-DETR

IV. Сравнение

В данном разделе был проведен сравнительный анализ двух подходов к видеоаналитике посетителей розничного магазина: комбинации детектора YOLO с трекером ByteTrack и комбинации детектора RT-DETR с алгоритмами многообъектного трекинга DeepSORT/StrongSORT. Сравнение ориентировано на практическое применение в условиях реальной торговой точки и направлено на оценку устойчивости детекции и трекинга без использования дополнительной обучающей выборки.

Сравнение проводилось на видеоданных, полученных с камер видеонаблюдения магазина одежды. Видеопоследовательности характеризуются фиксированной точкой обзора, наличием зон входа и кассы, а также присутствием статичных объектов (манекенов), которые могут ошибочно детектироваться как люди. В рамках работы не использовалась ручная разметка данных, поэтому количественная оценка точности детекции не проводилась. Основное внимание уделялось качественной оценке устойчивости трекинга и практической пригодности архитектур.

Для анализа работы систем использовались стандартные критерии, применяемые в задачах многообъектного трекинга:

- **Устойчивость идентификаторов** — способность системы сохранять один и тот же идентификатор объекта на протяжении его присутствия в кадре;
- **Фрагментация траекторий** — количество разрывов одной траектории на несколько идентификаторов;
- **Ложные треки** — появление идентификаторов, связанных с кратковременными или ошибочными детекциями;
- **Вычислительная сложность** и скорость обработки видеопотока.

В теории многообъектного трекинга данные характеристики формализуются с помощью метрик **MOTA**, **IDF1**, **ID switches** и **fragmentation**. В рамках работы данные метрики рассматривались на качественном уровне — через визуальный анализ траекторий и поведения идентификаторов во времени, что является распространённым подходом при отсутствии эталонной разметки[19].

Подход YOLO + ByteTrack продемонстрировал высокую скорость обработки видеопотока и стабильную работу в режиме, близком к реальному времени. Однопроходная архитектура детектора YOLO и простая процеду-

ра ассоциации в ByteTrack обеспечивают низкие вычислительные затраты. Использование детекций с низкой уверенностью снижает количество пропусков объектов и уменьшает фрагментацию траекторий при кратковременных перекрытиях. Однако отсутствие анализа внешнего вида объектов приводит к увеличению числа смен идентификаторов в сложных сценах, что негативно отражается на устойчивости трекинга при длительных перекрытиях.

Комбинация RT-DETR с алгоритмами DeepSORT/StrongSORT демонстрирует более устойчивое сохранение идентификаторов за счёт использования признаков внешнего вида объектов. Это позволяет снизить количество переключений идентификаторов и повысить согласованность траекторий во времени, что положительно сказывается на показателях IDF1 и уменьшает фрагментацию треков. В то же время трансформерная архитектура детектора и дополнительные нейросетевые модели для re-identification увеличивают вычислительную нагрузку и снижают скорость обработки кадров по сравнению с подходом YOLO + ByteTrack.

Таблица 1 – сравнение архитектур детекции и трекинга

Характеристика	**YOLO + ByteTrack**	**RT-DETR + Deep/StrongSORT**
Тип детектора	Сверточный	Трансформерный
Использование глобального контекста	Ограниченное	Полное
Анализ внешнего вида	Нет	Да
Устойчивость идентификаторов	Средняя	Высокая
Фрагментация траекторий	Низкая–средняя	Низкая
Вычислительные затраты	Низкие	Повышенные
Подходит для real-time	Да	Условно

Общее количество посетителей, зафиксированных системой, составило 14 для архитектуры YOLO + ByteTrack и 15 для архитектуры RT-DETR + StrongSORT/DeepSORT. Незначительное увеличение числа посетителей во втором случае может указывать на более устойчивое обнаружение и удержание объектов в кадре при использовании трансформерного детектора и трекера, учитывающего признаки внешнего вида. Это особенно важно в условиях перекрытий и временных исчезновений объектов из поля зрения камеры, характерных для торговых залов.

Анализ длительности присутствия посетителей в кадре показал заметные различия между двумя подходами. В случае YOLO + ByteTrack максимальное зафиксированное время пребывания составило около 38 секунд, тогда как для RT-DETR + StrongSORT/DeepSORT данный показатель достигал 58 секунд. Более высокая максимальная и средняя длительность присутствия во втором случае свидетельствует о меньшей фрагментации траекторий и более стабильном сохранении идентификаторов на протяжении всего времени нахождения посетителя в кадре. В реализации на основе YOLO + ByteTrack наблюдается большее количество треков с длительностью в диапазоне от 10 до 20 секунд, что может быть следствием разрывов одной реальной траектории на несколько коротких идентификаторов.

Распределение времени пребывания посетителей по функциональным зонам магазина в целом соответствует ожидаемым сценариям поведения покупателей. В обоих подходах основная часть времени фиксируется в торговой зоне, при этом переходы между торговым залом и кассовой зоной корректно отражаются в статистике. Однако в случае YOLO + ByteTrack у ряда посетителей наблюдаются кратковременные интервалы пребывания в зоне кассы продолжительностью менее одной-двух секунд, что может быть связано с нестабильной ассоциацией идентификаторов при пересечении зон. В реализации на основе RT-DETR и StrongSORT/DeepSORT временные интервалы по зонам выглядят более согласованными и непрерывными, включая длительные периоды нахождения у кассы, что указывает на более устойчивую идентификацию посетителей при перемещении между зонами.

Несмотря на отсутствие прямого вычисления формальных метрик многообъектного трекинга, таких как MOTA или IDF1, качественный анализ итоговой статистики позволяет сделать косвенные выводы о характере работы алгоритмов. Подход YOLO + ByteTrack демонстрирует высокую вычислительную эффективность и пригодность для задач реального времени, однако характеризуется большей чувствительностью к перекрытиям и изменениям позы объектов, что приводит к увеличению фрагментации треков. В свою очередь, комбинация RT-DETR с алгоритмами StrongSORT/DeepSORT обеспечивает более стабильное сохранение идентификаторов и формирование длинных непрерывных траекторий за счёт использования признаков внешнего вида, но требует больших вычислительных ресурсов.

Таким образом, проведённое сравнение показывает, что выбор архитектуры должен определяться приоритетами прикладной задачи. Для сценариев, где ключевым требованием является скорость обработки видеопотока и минимальные вычислительные затраты, более целесообразным является использование подхода YOLO + ByteTrack. В задачах, ориентированных на детальный анализ поведения посетителей и высокую устойчивость трекинга в сложных сценах, предпочтение следует отдавать архитектуре RT-DETR в сочетании с алгоритмами StrongSORT/DeepSORT.

На рисунках 10 – 11 приведена проанализированная выше статистика, рассчитанная для входного ролика продолжительностью 1 минута.

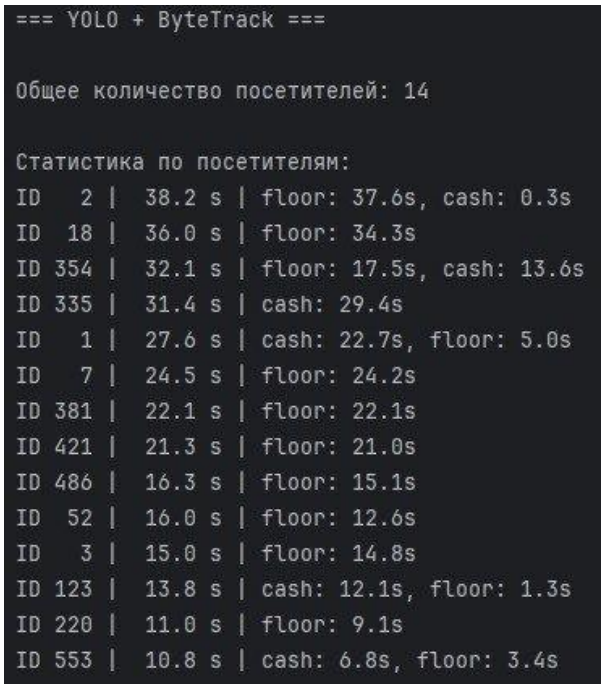


Рис. 10. Статистика анализа посетителей с помощью метода YOLO + ByteTrack

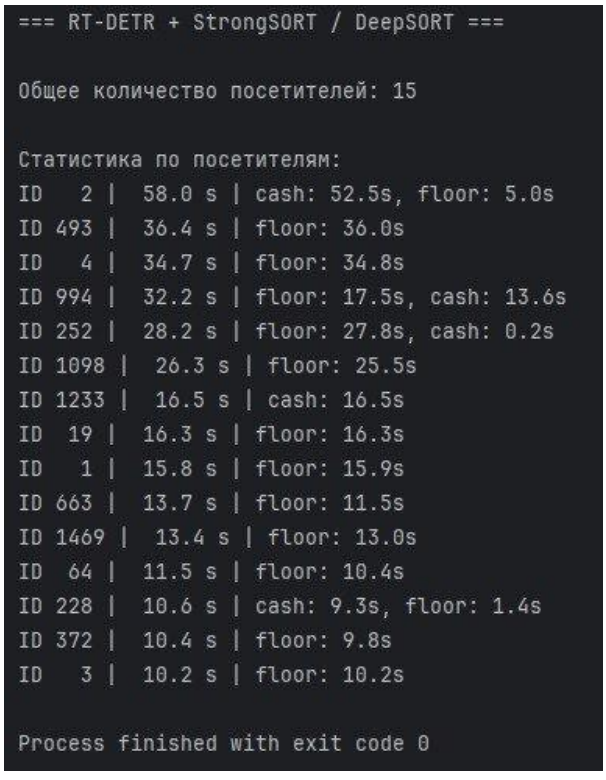


Рис. 11. Статистика анализа посетителей с помощью метода RT-DETR + StrongSORT/DeepSORT

V. Заключение

В рамках данной работы была рассмотрена задача видеоаналитики посетителей розничного магазина на основе методов компьютерного зрения и многообъектного трекинга. Основное внимание уделялось анализу поведения готовых нейросетевых архитектур в условиях реальной эксплуатации без дополнительного обучения и ручной разметки данных. В качестве входных данных использовались видеозаписи с камер видеонаблюдения торгового зала, характеризующиеся фиксированной точкой обзора, наличием функциональных зон и типичными для розничной торговли сценариями поведения посетителей.

В ходе работы были реализованы и исследованы два подхода к детекции и трекингу людей: комбинация детектора YOLO с алгоритмом ByteTrack и комбинация трансформерного детектора RT-DETR с алгоритмами StrongSORT/DeepSORT. Для обоих подходов была разработана единая программная система, обеспечивающая обнаружение посетителей, отслеживание их перемещений, агрегацию времени пребывания по зонам магазина и формирование итоговой статистики. Дополнительно были реализованы механизмы фильтрации кратковременных треков и подавления статичных объектов, что позволило повысить корректность аналитических результатов.

Сравнительный анализ показал, что архитектура YOLO + ByteTrack обеспечивает высокую скорость обработки видеопотока и стабильную базовую аналитику поведения посетителей при относительно низких вычислительных затратах. Данный подход хорошо подходит для систем, ориентированных на работу в режиме, близком к реальному времени, и может эффективно применяться в условиях ограниченных аппаратных ресурсов. В то же время использование исключительно геометрических критериев ассоциации объектов приводит к повышенной фрагментации траекторий в сложных сценах с перекрытиями.

Архитектура RT-DETR в сочетании с алгоритмами StrongSORT/DeepSORT продемонстрировала более устойчивое сохранение идентификаторов и формирование длинных непрерывных траекторий за счёт учёта признаков внешнего вида объектов и глобального контекста сцены. Это положительно сказывается на качестве анализа поведения посетителей и распределения времени по зонам магазина, однако сопровождается увеличением вычислительной сложности и снижением скорости обработки видеопотока.

Полученные результаты подтверждают, что выбор архитектуры для систем видеоаналитики должен определяться требованиями конкретной прикладной задачи, балансом между скоростью и качеством трекинга, а также доступными вычислительными ресурсами. Разработанный подход и проведённое сравнение могут быть использованы в качестве основы для построения и дальнейшего развития интеллектуальных систем анализа поведения посетителей в розничной торговле. В дальнейшем работа может быть расширена за счёт внедрения количественных метрик оценки качества трекинга, адаптации моделей под конкретные условия съёмки и интеграции дополнительных аналитических модулей.

Литература

[1] Алексеев, И. Б. Исследование возможности классификации человеческих действий / И. Б. Алексеев, П. Е. Злакоманов // *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. 112–117. – EDN WANSCF.

[2] Карякин, А. В. Исследование задачи детектирования человека с помощью компьютерного зрения / А. В. Карякин // *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. 61–64. – EDN MOSHUE.

[3] Крапухина, Н. В. Метод для извлечения необходимых и актуальных данных из видеопотока высокой информационной плотности / Н. В. Крапухина, Н. В. Каменов // *Системный анализ и информационные технологии (САИТ-2017)* : Сборник трудов Седьмой Международной конференции, Светлогорск, 13–18 июня 2017 года. – Светлогорск: Федеральный исследовательский центр «Информатика и управление» Российской академии наук, 2017. – С. 638–643. – EDN ZGPTGD.

[4] Методология интеллектуальной навигации для управления автономными подвижными объектами на основе триангуляции Делоне / С. О. Крамаров, О. Ю. Митясова, И. О. Темкин, В. В. Храмов // *Горный информационно-аналитический бюллетень (научно-технический журнал)*. – 2021. – № 2. – С. 87–98. – DOI 10.25018/0236-1493-2021-2-0-87-98. – EDN KKOOOG.

[5] 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, pp. 779–788.

[6] Гужва, Н. С. Алгоритмы идентификации светофоров на основе нейронных сетей в мультикамерных системах помощи водителю / Н. С. Гужва, Р. Н. Садеков // *Гироскопия и навигация*. – 2024. – Т. 32, № 3(126). – С. 47–65. – EDN IGOZNL.

[7] Dollar P., Wojek C., Schiele B., Perona P. Pedestrian Detection: An Evaluation of the State of the Art. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2012.

[8] Zhang Y., Sun P., Jiang Y., Yu D. ByteTrack: Multi-Object Tracking by Associating Every Detection Box. *European Conference on Computer Vision (ECCV)*, 2022, pp. 1–18.

[9] Bewley A., Ge Z., Ott L., Ramos F., Upcroft B. Simple Online and Realtime Tracking. *IEEE International Conference on Image Processing (ICIP)*, 2016, pp. 3464–3468.

[10] Wojke N., Bewley A., Paulus D. Simple Online and Realtime Tracking with a Deep Association Metric. *IEEE International Conference on Image Processing (ICIP)*, 2017, pp. 3645–3649.

[11] Du Y., Zhao Z., Song Y. et al. StrongSORT: Make DeepSORT Great Again. *arXiv preprint* arXiv:2202.13514, 2022.

[12] Lin T.-Y., Maire M., Belongie S. et al. Microsoft COCO: Common Objects in Context. *European Conference on Computer Vision (ECCV)*, 2014, pp. 740–755.

[13] Carion N., Massa F., Synnaeve G. et al. End-to-End Object Detection with Transformers. *European Conference on Computer Vision (ECCV)*, 2020, pp. 213–229.

[14] Zhang Y., Sun P., Jiang Y. et al. RT-DETR: Real-Time Detection Transformer. *arXiv preprint* arXiv:2304.08069, 2023.

[15] Kang K., Ouyang W., Li H. et al. Object Detection from Video Tubes with Convolutional Neural Networks. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[16] Chen L., Papandreou G., Kokkinos I. et al. DeepLab: Semantic Image Segmentation with Deep Convolutional Nets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2018.

[17] Luiten J., Osep A., Dendorfer P. et al. HOTA: A Higher Order Metric for Evaluating Multi-Object Tracking. *International Journal of Computer Vision*, 2021, vol. 129, pp. 548–578.

[18] Redmon J., Farhadi A. YOLOv3: An Incremental Improvement. *arXiv preprint* arXiv:1804.02767, 2018.

[19] Bernardin K., Stiefelhagen R. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *EURASIP Journal on Image and Video Processing*, 2008.

ГРНТИ 28.23.15
ASJC 1700
OECD 1.2

Регрессионная модель для предсказания точки фиксации взгляда пользователя на экране по данным RGB-видеопотока

А. А. Журавлева
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m25168282@edu.misis.ru

***Аннотация*—задача предсказания точки фиксации взгляда пользователя на экране по данным RGB-видеопотока в режиме реального времени является актуальной для систем человеко-машинного взаимодействия, интерфейсов виртуальной и дополненной реальности, а также вспомогательных технологий. В рамках работы предложена регрессионная нейросетевая модель, принимающая на вход изображения области глаз пользователя и формирующая на выходе координаты точки фиксации взгляда в экранной системе координат. В отличие от задач классификации направления взгляда в дискретных направлениях, в данной работе рассматривается непрерывная регрессия экранных координат. Для решения задачи используется сверточная нейронная сеть с остаточными блоками и механизмом пространственного внимания, повышающим устойчивость модели к вариациям входных данных. Обучение модели осуществляется на собственном размеченном датасете, сформированном в процессе индивидуальной калибровки пользователя. В работе приведено описание структуры набора данных, архитектуры нейронной сети, процесса обучения и результатов экспериментальной оценки. Полученные результаты подтверждают работоспособность предложенного подхода и возможность его применения в системах реального времени.**

Ключевые слова — глубокое обучение, видеопоток, компьютерное зрение, направление взгляда, нейронные сети, регрессионная модель, сверточная нейронная сеть, человеко-машинный интерфейс

I. Введение

Методы компьютерного зрения и машинного обучения широко применяются в задачах анализа визуальной информации, поступающей в виде непрерывного видеопотока, включая системы наблюдения, навигации и интерактивные пользовательские интерфейсы [1]. Одной из прикладных задач данного направления является неинвазивное отслеживание взгляда человека, позволяющее определить, на какую область экрана направлено визуальное внимание пользователя, без применения специализированных аппаратных трекеров.

В отличие от классических айтрекеров, основанных на инфракрасной подсветке и специализированных датчиках, программные решения, использующие RGB-камеры, отличаются меньшей стоимостью и большей доступностью. Однако такой подход предъявляет повышенные требования к алгоритмам обработки изображений, устойчивости моделей к шуму видеопотока и вариациям условий съёмки. Современные исследования показывают, что сверточные нейронные сети эффективно извлекают пространственные признаки из изображений и позволяют решать задачи регрессии непрерывных величин по визуальным данным [2].

Качество предсказания точки фиксации взгляда существенно зависит от корректного выделения области глаз и стабилизации входных изображений, что подчёркивается в работах, посвящённых анализу качества визуальных данных и предварительной обработке изображений в системах компьютерного зрения [3, 4]. При этом важным аспектом является выбор архитектуры модели и механизмов, позволяющих акцентировать внимание на наиболее информативных участках изображения, что активно исследуется в работах по анализу архитектур нейронных сетей и механизмов внимания [5].

В рамках данной работы рассматривается задача регрессионного предсказания экранных координат точки фиксации взгляда пользователя по изображениям области глаз, получаемым из RGB-видеопотока. Используется собственный датасет, сформированный в процессе калибровки пользователя, содержащий пары «изображение глаз — координаты точки взгляда на экране», опубликованный в открытом доступе на платформе HuggingFace Datasets [6]. Такой подход позволяет адаптировать модель под конкретные параметры экрана и условия эксплуатации.

Целью работы является разработка и экспериментальная оценка нейросетевой модели, способной в реальном времени предсказывать координаты точки фиксации взгляда на экране на основе изображений области глаз.

II. Набор данных

Основным источником данных для обучения модели является RGB-видеопоток с веб-камеры, из которого извлекаются последовательные кадры области глаз пользователя. Использование видеопотока в качестве входных данных является стандартным подходом в современных системах компьютерного зрения и анализа визуальной информации [7].

Набор данных формируется в процессе индивидуальной калибровки пользователя: в заранее определённые моменты времени пользователь фиксирует взгляд на заданных точках экрана, координаты которых известны системе. Для каждого момента фиксации сохраняется изображение области

глаз и соответствующая пара экранных координат (X, Y), задающих положение точки фиксации взгляда. Такой формат представления данных соответствует классической постановке задачи регрессии в компьютерном зрении [8].

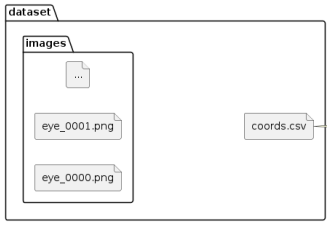


Рис. 1. Структура набора данных изображений глаз и координат взгляда

На рисунке 1 представлена логическая структура сформированного набора данных, включающая изображения области глаз и соответствующие им координаты точки фиксации взгляда на экране.

Данная схема иллюстрирует взаимосвязь между визуальными данными и целевыми значениями, используемыми при обучении регрессионной модели.

Данные организованы в виде набора изображений в формате PNG и таблицы аннотаций в формате CSV, где каждой строке соответствует пара «имя изображения — координаты точки фиксации». Пример содержимого файла аннотаций — на рисунке 2.

Рис. 2. Пример данных, содержащихся в наборе данных

Предварительная обработка изображений является необходимым этапом для повышения устойчивости модели к вариациям условий съёмки, таким как изменения освещения, шум изображения и незначительные движения головы пользователя.

Последовательность ключевых этапов обработки видеопотока и формирования области глаз показана на рисунке 3.

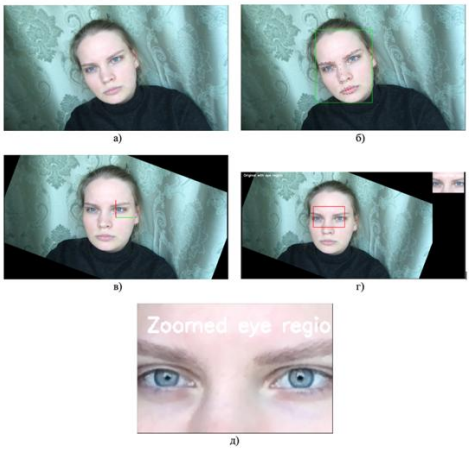


Рис. 3. Этапы обработки видеопотока: (а) исходный кадр, (б) детекция лица и ключевых точек, (в) поворот изображения, (г) выделенная область глаз, (д) результат.

Методы нормализации, геометрической коррекции и приведения изображений к единому формату широко применяются в задачах компьютерного зрения [9].

В данной работе предобработка включает зеркальное отражение кадров, детекцию лица и ключевых точек (facial landmarks), а также компенсацию наклона головы на основе геометрии расположения глаз. На основании координат ключевых точек формируются стабилизированные области интереса, охватывающие область глаз, которые далее масштабируются и приводятся к фиксированному разрешению. Такая стабилизация входных данных позволяет снизить влияние положения головы и повысить воспроизводимость признаков, используемых моделью.

III. Архитектура модели

При построении модели предсказания направления взгляда по изображениям глаз был выбран подход, основанный на сверточных нейронных сетях (Convolutional Neural Networks, CNN), поскольку они хорошо зарекомендовали себя в задачах извлечения пространственных признаков из изображений и позволяют моделям эффективно обрабатывать визуальные входные данные [10].

Одной из проблем глубоких CNN является ухудшение сходимости при увеличении глубины сети, что связано с исчезновением градиента в глубоких слоях. Для решения этой проблемы широко используются остаточные блоки (residual blocks), впервые введённые в архитектуре ResNet. Остаточные блоки добавляют прямые «shortcut»-связи, которые облегчают распространение градиента и позволяют создавать более глубокие и устойчивые модели [11].

В качестве отправной точки для разработки архитектуры была взята предобученная на базе ImageNet модель ResNet-18, адаптированная для задачи регрессии координат. Предобученная модель позволяет использовать знания, полученные при обучении на большом наборе реальных изображений, что улучшает начальную инициализацию весов и ускоряет сходимость.

На основе этого был разработан собственный вариант архитектуры, названный EyeGazeNetV3. Входным слоем модели является блок свёрток и нормализаций, за которым следуют несколько последовательных остаточных блоков, обеспечивающих извлечение глубоких признаков. Внутри этих блоков применяются свёртки с функциями активации ReLU и пакетная нормализация, что повышает стабильность обучения.

Особое место в архитектуре занимает механизм пространственного внимания (spatial attention), позволяющий модели фокусироваться на наиболее информативных участках изображения. Attention-механизм реализован как последовательность сверточных слоёв, которые формируют карту внимания, коэффициенты которой находятся в диапазоне $[0, 1]$ и отражают важность каждой позиции на признаковом тензоре. Карта внимания умножается поэлементно на признаки, что усиливает значимые области изображения (например, область глаз) и подавляет фоновые шумы [12].

Заключительная часть сети представляет собой регрессионную «голову»: после глобального усреднения признаков (Adaptive Average Pooling) используется несколько полносвязных слоёв, завершающихся выходом из двух нейронов, соответствующих координатам взгляда по осям X и Y. Для улучшения устойчивости сходимости при обучении используется функция потерь Smooth L1Loss, также известная как Huber-loss, которая проявляет умеренное поведение по отношению к выбросам в данных и обеспечивает более стабильную оптимизацию, чем Mean Squared Error (MSE) при наличии шумных примеров [13].

Схематическое представление архитектуры модели EyeGazeNetV3 приведено на рисунке 4.

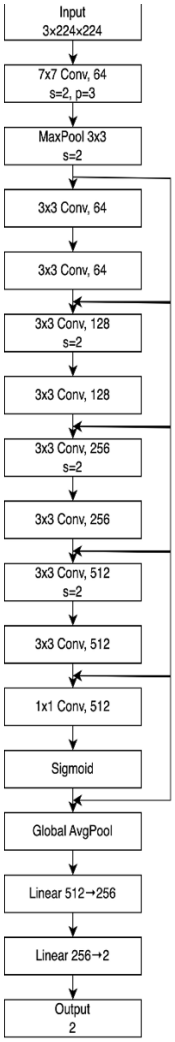


Рис. 4. Архитектура модели EyeGazeNetV3 с остаточными блоками и механизмом внимания

IV. Результаты и анализ

В ходе экспериментов с архитектурами моделей EyeGazeNet были получены следующие результаты. Основное внимание уделялось сравнению различных конфигураций сверточных блоков, функций потерь и стратегий оптимизации.

Для оценки качества предсказаний использовались стандартные метрики: средняя абсолютная ошибка (MAE), среднеквадратическая ошибка (RMSE), стандартное отклонение и евклидова ошибка, а также коэффициент детерминации R^2. Выбор метрик обоснован их популярностью в задачах регрессии координат взгляда и демонстрирует чувствительность к как систематическим, так и случайным отклонениям предсказаний [14].

A. Сравнение ключевых метрик

В Таблице 1 представлены основные метрики, полученные для нескольких архитектур моделей, начиная с базового варианта ResNet18 и заканчивая финальной версией модели EyeGazeNetV4 с использованием механизма внимания и тонкой настройки (fine-tuning).

ТАБЛИЦА I. Сводные метрики качества моделей по экспериментам

	ResNet18	EyeGazeNet V2	EyeGazeNet V3	EyeGazeNet V4 (fine-tuned)
MAE X (px)	117.04	133.77	18.16	17.13
MAE Y (px)	212.86	227.73	19.19	18.06
Euclidean Error (px)	247.68	287.84	29.46	27.79
R^2 X	0.878	0.800	0.996	0.996
R^2 Y	-0.065	-0.033	0.990	0.991

Как видно из Таблицы 1, базовые архитектуры на основе ResNet18 и упрощённой сети демонстрируют относительно высокие значения ошибок и низкие коэффициенты детерминации, особенно по вертикальной оси Y. Такие результаты свидетельствуют о недостаточной способности моделей учиться сложным зависимостям между изображением глаз и истинной координатой взгляда. При этом существенное улучшение достигается в модификациях EyeGazeNet V3 и V4: обе версии показывают резкое уменьшение абсолютной ошибки (< 20 px) и R^2 близкие к 1, что указывает на высокую согласованность предсказаний с истинными значениями.

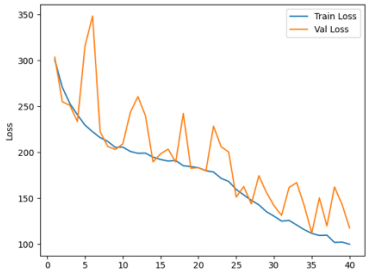
Следует отметить, что коэффициент детерминации R^2 является чувствительной метрикой для задач регрессии: значения, близкие к 1, говорят о том, что модель объясняет большую часть изменчивости данных, тогда как отрицательные значения (как в первых версиях) указывают на неспособность модели адекватно описывать зависимость между входными изображениями и целевыми координатами.

Анализ таблицы показывает, что использование упрощённых архитектур и L1-подобных функций потерь обеспечило более стабильное обучение и снижение MAE и RMSE по сравнению с исходной ResNet18 [13]. Внедрение механизма attention позволило улучшить интерпретируемость модели, но вертикальная точность оставалась ограниченной, что связано с распределением данных и особенностями обучения.

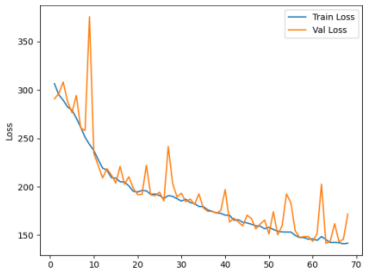
B. Динамика обучения моделей

Для каждого варианта архитектуры были построены графики динамики процесса обучения и валидации, включающие значения train_loss, validation_loss (рисунок 5).

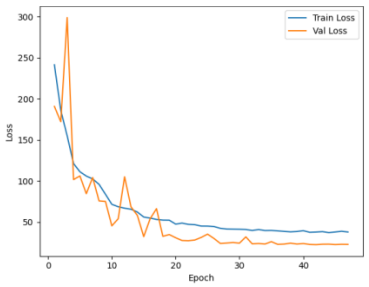
а



б



в



г

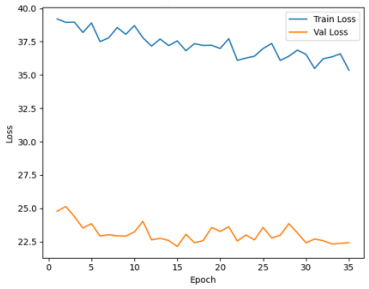


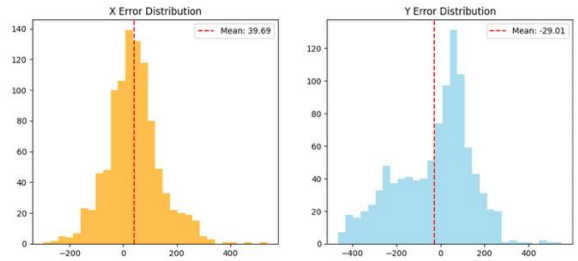
Рисунок 5 – Динамика обучения моделей: (а) ResNet18, (б) EnhancedEyeGazeNet v2, (в) EyeGazeNet V3, (г) EyeGazeNet V4 (fine-tuned)

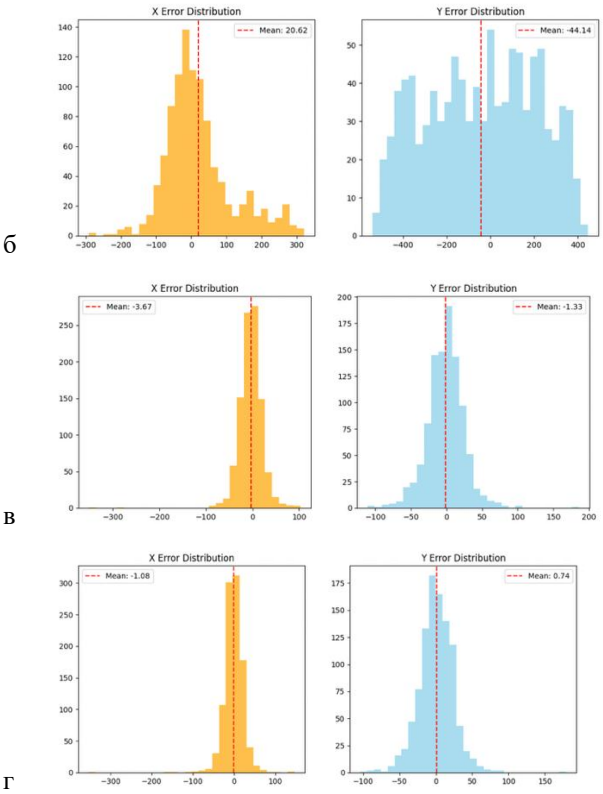
На рисунке 5 видно, что первые две модели демонстрируют выраженный разрыв между кривыми обучения и валидации, что типично для переобучения на ограниченном датасете. В отличие от них, модели EyeGazeNet V3 и V4 демонстрируют более устойчивую тенденцию к снижению ошибок на валидации и менее выраженное расхождение между train и val loss, что указывает на лучшую обобщающую способность при прочих равных.

C. Распределение ошибок

Для каждой из ключевых моделей построены распределения ошибок по горизонтальной (X) и вертикальной (Y) осям (рисунок 6), что позволяет визуально оценить характер отклонений предсказаний от истинных значений.

а





б

в

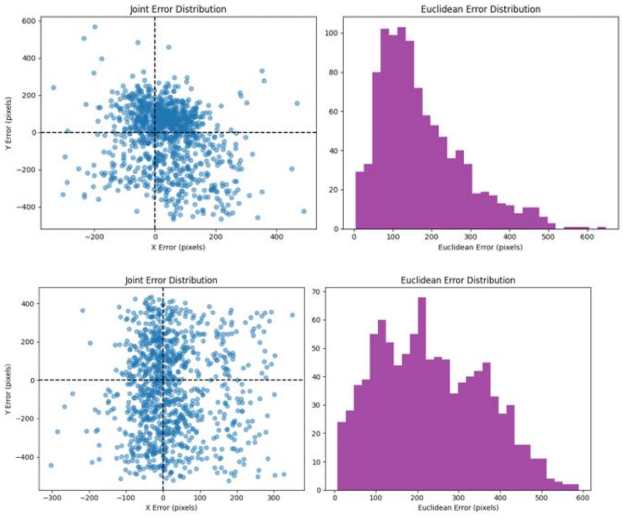
г

Рисунок 6 – Гистограммы распределения ошибок

Распределение ошибок первых моделей характеризуется широким разбросом и смещёнными пиками, особенно по оси Y, что свидетельствует о систематических ошибках. Напротив, модели EyeGazeNet V3 и V4 имеют более узкие распределения со сконцентрированными значениями около нуля, что логично коррелирует с низкими значениями MAE в Таблице 1.

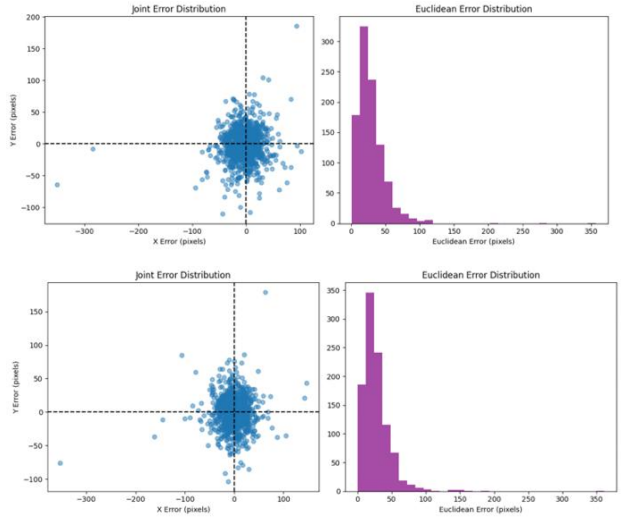
D. Визуализация ошибок предсказания

Дополнительно были построены графики рассеяния ошибок и распределения модуля ошибки для моделей EyeGazeNet V3 и V4 (рисунок 7), что позволяет оценить плотность и характер отдельных отклонений.



а

б



в

г

Рисунок 7 – Анализ погрешностей модели: (a) диаграмма рассеяния X-Y (V3), (b) распределение модуля ошибки (V3), (c) диаграмма рассеяния X-Y (V4), (d) распределение модуля ошибки (V4)

На рисунке 7 видно, что большинство точек предсказаний сконцентрировано вблизи истинных значений (центр координат), особенно для модели V4, где плотность рассеяния ошибок выше в узкой области. Гистограммы модульной ошибки подтверждают устойчивость модели: большая часть предсказаний лежит в диапазоне до 30 px, а редкие выбросы относятся к экстремальным условиям (например, движения глаз пользователя во время съёмки, шум видеопотока и т.п.).

V. Заключение

В данной работе была разработана и апробирована регрессионная нейросетевая модель предсказания направления взгляда в реальном времени на основе RGB-видеопотока. Для решения поставленной задачи использовался собственный размеченный набор данных и архитектура сверточной нейронной сети с остаточными блоками и встроенным механизмом пространственного внимания, что согласуется с современными подходами к анализу визуальных данных в задачах компьютерного зрения [1, 15].

Проведённые эксперименты показали, что модели с residual-блоками и attention-механизмом (EyeGazeNetV3 и её тонкая настройка EyeGazeNetV4) существенно превосходят базовые архитектуры по ключевым метрикам качества предсказаний, включая среднюю абсолютную ошибку и коэффициент детерминации. Итоговые значения MAE < 20 пикселей по обеим осям и $R^2 > 0{,}99$ подтверждают приемлемый уровень точности для прикладных задач неинвазивного отслеживания взгляда, что находится в рамках требований к регрессионным задачам на изображениях [16].

Несмотря на положительные результаты, остаются ограничения, связанные с однородностью датасета и чувствительностью к условиям освещения и вариациям положения пользователя. В будущем планируется расширение набора данных, исследование методов адаптивного обучения и интеграция модели в полноценные системы человеко-машинного взаимодействия, где оценка качества входных

изображений и компенсация искажений могут быть критическими для стабильности предсказаний.

Литература

[1] Чернов, Н. П. Разумный, А. С. Кожаринов [и др.] // Информационные технологии и вычислительные системы. – 2017. – № 4. – С. 71-82. – EDN ZXKJIJ.

[2] 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.

[3] Антипов, И. И. Исследование возможности классификации мусора 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 декабря 2023 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2023. – С. 16-21. – EDN TRQDLA.

[4] Фомина, А. А. Поиск ключевых точек на изображении лица / А. А. Фомина // 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. 133-139. – EDN VAETXI.

[5] Зайцева, Е. В. Сравнение архитектур нейронных сетей для задачи сегментации изображений / Е. В. Зайцева, В. К. Казанков // Актуальные вопросы развития современной цифровой среды : сборник статей по материалам научно-технической конференции молодых ученых, Москва, 14–16 апреля 2021 года. – Волгоград: ИП ЧЕРНЯЕВА ЮЛИЯ ИГОРЕВНА (Издательский дом "Сириус"), 2021. – С. 403-409. – EDN VQSQWL.

[6] LackOfImagination. Eye Gaze RGB Dataset [Электронный ресурс] // Hugging Face. – URL: https://huggingface.co/datasets/lackOfImagination/eye-gaze-rgb-dataset.

[7] Компьютерное зрение в 2024 году: Главные задачи и направления // Habr.com / MaxRokatansky. — 2024. — URL: https://habr.com/ru/companies/otus/articles/810207/.

[8] Исследование возможностей компьютерного зрения в мобильных браузерах // Cyberleninka.ru / Череповский М. — 2024. — Журнал «Вестник науки». — URL: https://cyberleninka.ru/article/n/issledovanie-vozmozhnostey-kompyuternogo-zreniya-v-mobilnyh-brauzerah.

[9] Алгоритмы обработки изображений для систем компьютерного зрения в мехатронике // Pandia.org. — URL: https://pandia.org/text/77/295/32005.php.

[10] Сверточные нейронные сети и их применение в задачах компьютерного зрения // Habr.com / А. Иванов. — 2020. — URL: https://habr.com/ru/articles/424202/.

[11] Residual Neural Networks (ResNet) и остаточные блоки // Habr.com / П. Петров. — 2020. — URL: https://habr.com/ru/articles/428364/.

[12] Использование attention в сверточных нейронных сетях для компьютерного зрения // Towards Data Science / J. Doe. — 2020. — URL: https://towardsdatascience.com/attention-in-cnn-for-computer-vision-624635491b64.

[13] Smooth L1 Loss (Huber Loss) — описание и применение // PyTorch Documentation. — URL: https://pytorch.org/docs/stable/generated/torch.nn.SmoothL1Loss.html.

[14] Кукурхоев, А. М. Функция потерь MSE и MAE / А. М. Кукурхоев // НАУКА, ОБРАЗОВАНИЕ, ИННОВАЦИИ: АКТУАЛЬНЫЕ ВОПРОСЫ и современные АСПЕКТЫ : сборник статей XV Международной научно-практической конференции в 2 частях, Пенза, 29 декабря 2022 года. Том Часть 1. – Пенза: Наука и Просвещение (ИП Гуляев Г.Ю.), 2022. – С. 85-86. – EDN IOCOZC.

[15] R. J. Williams, D. Zipser, “A learning algorithm for continually running fully recurrent neural networks”, Neural Computation, vol. 1, no. 2, pp. 270–280, 1989.

[16] A. Krizhevsky, I. Sutskever, G. Hinton, “ImageNet classification with deep convolutional neural networks”, Advances in Neural Information Processing Systems, 2012.

ГРНТИ: 28.00.00, 28.23.00, 28.23.15
ASJC: 1702, 1707
OECD: 1.2, 1.2.2, 1.2.3

Детекция объектов дорожной сцены на изображениях с применением сверточных нейронных сетей

У. Р. Зейналов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2107072@edu.misis.ru

Аннотация — в работе рассматривается задача детекции объектов дорожной сцены на цифровых изображениях с использованием методов компьютерного зрения и сверточных нейронных сетей. Актуальность исследования обусловлена необходимостью автоматизации процессов восприятия визуальной информации в интеллектуальных транспортных системах и системах технического зрения. В рамках исследования был сформирован и размечен набор данных изображений дорожной сцены с применением инструмента CVAT.AI, после чего выполнено обучение и тестирование нейросетевых моделей для детектирования и классификации объектов. Проведен сравнительный анализ результатов работы моделей по ряду количественных показателей. Полученные результаты демонстрируют эффективность применения методов глубокого обучения для задач распознавания объектов дорожной сцены и подтверждают целесообразность их использования в прикладных системах компьютерного зрения.

Ключевые слова: анализ изображений, детекция, компьютерное зрение, обработка изображений, распознавание объектов, сверточные нейронные сети, Faster R-CNN, Yolo8.

I. Введение

Развитие методов компьютерного зрения является одним из приоритетных направлений современных исследований в области искусственного интеллекта и прикладной информатики [1, 2]. Особое значение приобретают задачи автоматического анализа изображений дорожной сцены, поскольку они лежат в основе функционирования интеллектуальных транспортных систем, систем помощи водителю и автономных транспортных средств. Надежное распознавание объектов дорожной инфраструктуры, таких как транспортные средства, пешеходы и дорожные знаки, является необходимым условием повышения безопасности и эффективности дорожного движения [3–7].

Одним из наиболее распространенных классов моделей для решения задач детекции объектов являются двухстадийные методы, к которым относится архитектура Faster R-CNN. Данные модели обеспечивают высокую точность локализации объектов за счет поэтапного формирования у регионов интереса и последующей классификации, однако характеризуются высокой вычислительной сложностью и ограниченной скоростью работы. Это обстоятельство затрудняет их применение в системах реального времени, включая автономное вождение и робототехнические комплексы.

Альтернативным подходом являются одностадийные детекторы, представленные семейством моделей YOLO, в которых задача определения координат объектов и их классификации решается в рамках единой нейросетевой архитектуры [4–7]. Такие модели обеспечивают высокую скорость обработки изображений при сохранении приемлемого уровня точности. Современные версии YOLO демонстрируют конкурентоспособные результаты при анализе дорожных сцен и широко применяются в практических задачах компьютерного зрения.

В связи с выше изложенным актуальным является сравнительный анализ различных архитектур сверточных нейронных сетей при решении задачи детекции объектов дорожной сцены. Целью данной работы является исследование эффективности моделей YOLO и Faster R-CNN на собственном наборе данных изображений дорожной сцены, сформированном и размеченном с использованием инструмента CVAT.AI, а также оценка качества их работы.

II. Набор данных

Для проведения исследования был сформирован собственный набор данных изображений дорожной сцены. Сбор исходных данных осуществлялся путем видеосъемки городских улиц и прилегающих территорий с использованием камеры мобильного устройства. Видеозапись выполнялась с разрешением 1920×1080 пикселей при глубине цвета 24 бита. Съемка проводилась в различных условиях освещенности и при разной плотности дорожного движения — от предрассветных часов до середины дня, при регулярной смене локаций.

Полученные видеоматериалы были преобразованы в отдельные кадры с применением утилиты ffmpeg, что является распространенной практикой при

формировании датасетов для задач детекции объектов. Кадры извлекались с интервалом в две секунды, после чего осуществлялась их фильтрация по качеству: изображения с размытием или значительными артефактами исключались из дальнейшего рассмотрения.

Разметка изображений выполнялась вручную с использованием инструмента CVAT.AI, предназначенного для аннотирования данных в задачах компьютерного зрения и машинного обучения. Применение специализированных средств разметки позволяет повысить точность аннотаций и ускорить подготовку обучающих выборок [6, 8]. В процессе аннотирования для каждого объекта на изображении задавались ограничивающие прямоугольники и соответствующие классы.

В рамках разметки было принято решение объединять группы близко расположенных дорожных знаков в один объект, что обусловлено высокой плотностью их размещения в отдельных сценах. Данный подход не влияет на корректность исследования, поскольку в работе анализируется факт наличия дорожного знака, а не его конкретный тип.

Рис. 1 — Разметка одного дорожного знака

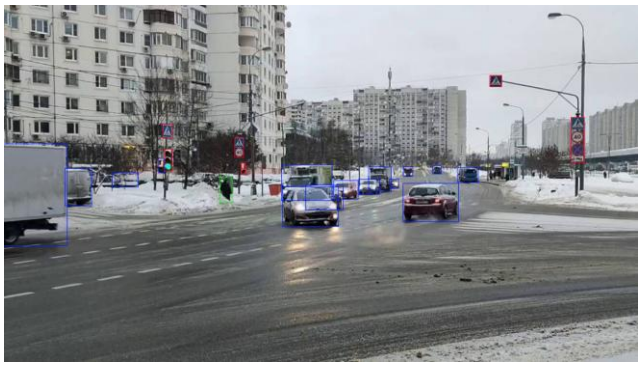


Рис. 2 — Разметка нескольких дорожных знаков

Дополнительной особенностью разметки стало наличие транспортных средств на дальнем плане, частично перекрытых снегом. Для оценки целесообразности их аннотирования были сформированы два экспериментальных поднабора данных, различающихся наличием разметки таких объектов. Анализ показал, что влияние данного фактора на итоговые метрики качества является незначительным, поэтому разметка дальних объектов выполнялась выборочно.

В результате разметки были выделены четыре класса объектов дорожной сцены: транспортные средства, пешеходы, дорожные знаки, а также отдельный класс для мотоциклов, самокатов и велосипедов. Выделение последнего класса обусловлено спецификой их движения, включая возможное перемещение по пешеходным зонам, и повышенной аварийной опасностью.

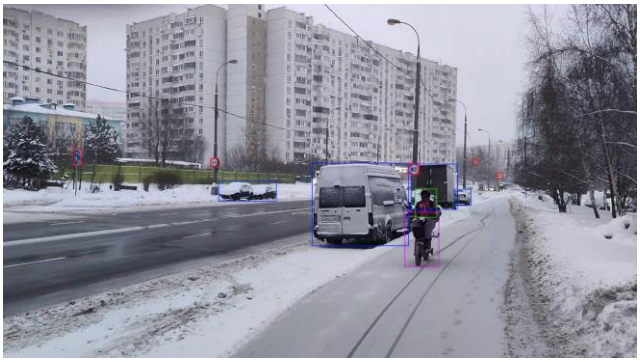


Рис. 3 — Мотоцикл на пешеходной части дороги

Итоговый набор данных включает 1109 изображений, разделенных на обучающую, валидационную и тестовую выборки в соотношении 70%, 15% и 15% соответственно [10, 11]. Такое разделение является стандартным при обучении и тестировании моделей глубокого обучения и позволяет получить объективную оценку качества их работы. Сформированный набор данных опубликован на платформе HuggingFace.

Сформированный набор данных опубликован на платформе HuggingFace и доступен по ссылке: https://huggingface.co/datasets/dumuzeyn/RODataset.

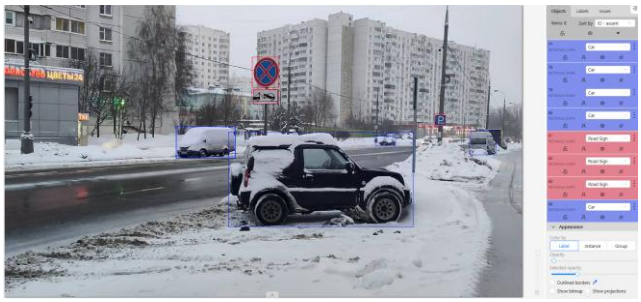


Рис. 4 — Разметка объектов дорожной сцены в CVAT.AI

III. Архитектура нейронной сети

В рамках данного исследования для решения задачи детекции объектов дорожной сцены использовались архитектуры сверточных нейронных сетей, относящиеся к классам одностадийных и двухстадийных детекторов объектов. Основной целью применения данных архитектур является автоматическое определение наличия объектов на изображении, их пространственной локализации и классификации по заданным категориям [5-8].

A. *Общие принципы работы сверточных нейронных сетей*

Сверточная нейронная сеть представляет собой параметризованную функцию:

$$f(x;\theta)\#(1)$$

где x — входное изображение, а θ — набор обучаемых параметров. Основу сети составляют

сверточные слои, выполняющие операцию свертки входных данных с набором фильтров:

$$y_{i,j,k} = \sum_{m,n,c} x_{i+m,j+n,c} \cdot w_{m,n,c,k} + b_k \#(2)$$

где w — веса фильтра, b — смещение, а y — выходное отображение признаков. Использование нелинейных функций активации и операций нормализации позволяет сети аппроксимировать сложные нелинейные зависимости в изображениях [6].

B. *Архитектуры семейства YOLO*

Модели семейства YOLO относятся к одностадийным детекторам объектов и реализуют принцип “обнаружение за один проход”. Изображение разбивается на сетку, для каждой ячейки которой нейронная сеть одновременно предсказывает координаты ограничивающих прямоугольников, вероятности наличия объекта и его класс [6].

Каждый предсказанный ограничивающий прямоугольник описывается вектором параметров:

$$(b_x, b_y, b_w, b_h, p_0, p_1, \dots, p_C) \#(3)$$

Где b_x, b_y — координаты центра прямоугольника,

b_w, b_h — его ширина и высота,

p_0 — вероятность наличия объекта,

$p_1, \dots, p_C$ — вероятности принадлежности к классам.

Функция потерь YOLO представляет собой взвешенную сумму ошибок локализации, классификации и уверенности детекции:

$$\mathrm{L} = \lambda_{loc} L_{loc} + \lambda_{conf} L_{conf} + \lambda_{cls} L_{cls} \#(4)$$

где каждая составляющая отвечает за соответствующий аспект качества предсказаний [6-8].

Архитектура YOLOv8 использует усовершенствованные модули извлечения признаков и оптимизированную головную часть сети, что повышает точность детектирования при высокой скорости. YOLOv11 является дальнейшим развитием данного подхода и ориентирована на более устойчивую работу в условиях перекрытия объектов [6, 7].

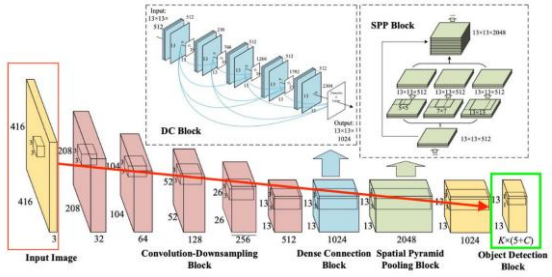


Рис. 5 — Разбиение изображение на блоки в YOLO

C. *Архитектура Faster R-CNN*

Faster R-CNN относится к классу двухстадийных детекторов и реализует поэтапный подход к обнаружению объектов. На первом этапе используется сеть предложений регионов (Region Proposal Network), которая формирует набор потенциальных областей, содержащих объекты [8].

Функция потерь RPN включает два компонента:

$$L_{RPN} = L_{cls} + L_{reg} \#(5)$$

Где L_{cls} — ошибка бинарной классификации региона,
L_{reg} — ошибка регрессии координат ограничивающего прямоугольника.

На втором этапе для каждого предложенного региона выполняется классификация и уточнение координат. В качестве базовой сверточной части Faster R-CNN в данной работе использовалась архитектура ResNet-50, основанная на использовании остаточных связей:

$$y = F(x) + x \#(6)$$

что позволяет эффективно обучать глубокие нейронные сети и снижать эффект деградации качества при увеличении числа слоёв [8, 11].

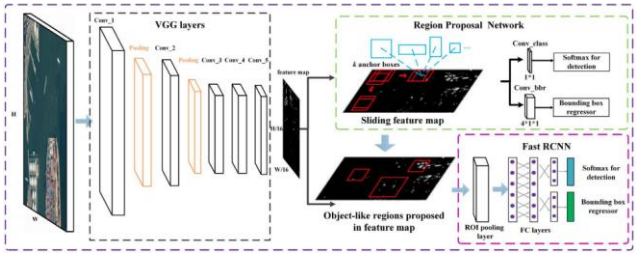


Рис. 6 — Схема работы Faster R-CNN

IV. Сравнение архитектурных подходов

Для оценки эффективности был проведён сравнительный анализ моделей YOLOv8, YOLOv11 и Faster R-CNN при решении задачи детекции объектов дорожной сцены. В качестве параметров сравнения использовались показатели точности (precision), полноты (recall), средняя точность при пороге IoU = 0.5 (mAP50), а также интегральный показатель mAP50-95, учитывающий качество локализации объектов при различных значениях порога перекрытия.

A. *Результаты обучения модели YOLOv8*

В таблице 1 представлены результаты обучения модели YOLOv8 на тестовой выборке в процессе 100 эпох обучения, а также значение следующих метрик:

- Precision — доля корректно обнаруженных объектов среди всех объектов, предсказанных моделью.
- Recall — доля корректно обнаруженных объектов среди всех объектов, присутствующих в разметке.
- mAP50 — среднее значение точности (Average Precision), вычисленное при фиксированном пороге перекрытия IoU = 0.5, характеризующее общее качество детекции при умеренных требованиях к локализации.
- mAP50–95 — усреднённая средняя точность, вычисленная по диапазону порогов IoU от 0.5 до 0.95 с шагом 0.05, отражающая устойчивость модели к точной локализации объектов.

Анализ результатов показывает устойчивый рост показателей mAP по мере обучения модели, и стабилизацию метрик precision и recall после 70-ой эпохи, что свидетельствует об эффективной сходимости и хорошей обобщающей способности модели. Относительно высокое значение precision говорит о том, что модель хорошо находит объекты на сцене, но невысокий показатель recall означает, что модель неправильно определяет класс для небольшого количества объектов.

Таблица 1 — Результаты YOLOv8 за 100 эпох

epoch	precision	recall	mAP50	mAP50-95
1	0,845040	0,133580	0,265900	0,137230
10	0,648580	0,540380	0,556400	0,286370
20	0,701450	0,576780	0,611610	0,331540
30	0,748830	0,583060	0,637110	0,352080
40	0,767400	0,586480	0,670320	0,376410
50	0,765650	0,613980	0,681870	0,385420
60	0,734590	0,665910	0,695750	0,395120
70	0,814080	0,612420	0,700160	0,406460
80	0,785560	0,637550	0,703250	0,422370
90	0,786480	0,651660	0,705360	0,422440
100	0,790260	0,623810	0,705000	0,423250

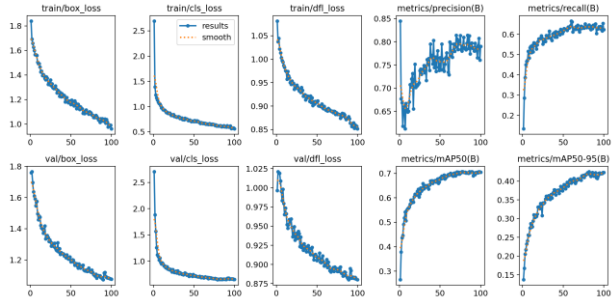


Рис. 7 — Графики метрик во время обучения YOLOv8

Матрица ошибок показывает, что модель YOLOv8 правильно находит объекты классов Car, Person, RoadSigns в 64% случаев, но в 69% случаев модель видит объект класса Car на фоне в месте, где его нет.

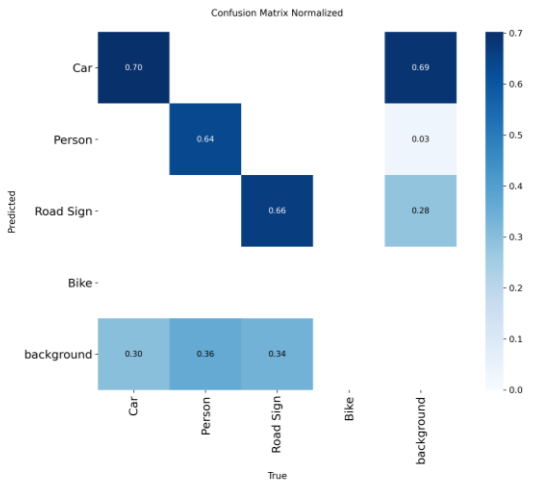


Рис. 8 — Нормализованная матрица ошибок

В датасете самым частым классом был Car и RoadSigns и никакой зависимости между ними не обнаружилось.

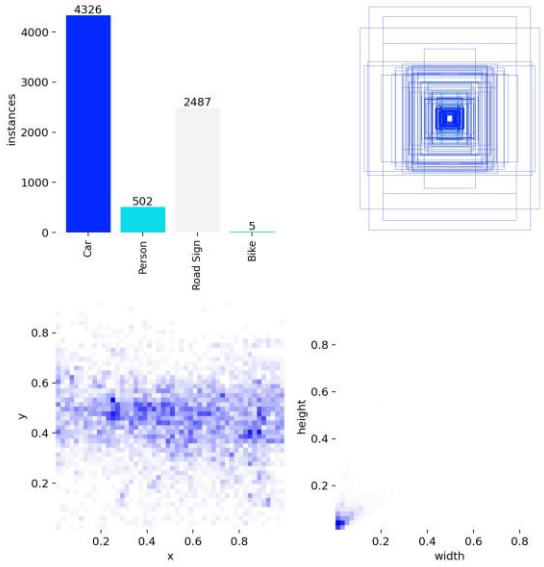


Рис. 9 — Распределение классов

B. *Сравнение YOLOv8 и YOLOv11*

Модель YOLOv11 была обучена на том же наборе данных с использованием аналогичных параметров обучения. В таблице 2 приведены итоговые показатели качества детектирования для моделей YOLOv8 и YOLOv11 на тестовой выборке.

Таблица 2 — Сравнение моделей YOLOv8 и YOLOv11

Модель	precision	recall	mAP50	mAP50-95
YOLOv8	0.79	0.62	0.71	0.42
YOLOv11	0.81	0.64	0.73	0.45

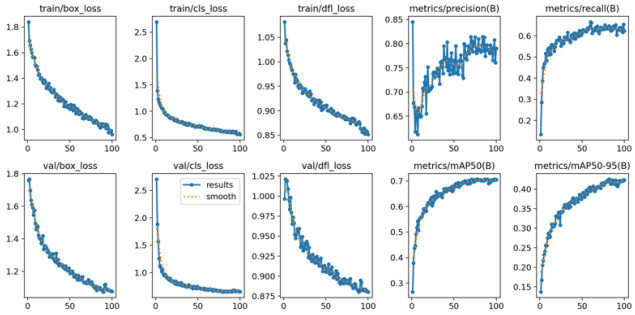


Рис. 10 — Графики метрик во время обучения YOLOv11

Модель YOLOv11 демонстрирует незначительное улучшение показателей precision и mAP по сравнению с YOLOv8, что можно объяснить более глубокой архитектурой и усовершенствованными механизмами агрегации признаков или погрешностью. При этом увеличение точности сопровождалось ростом вычислительной сложности, что увеличило время обработки почти в 1.87 раз.

Как видно из рисунка графики моделей YOLOv8 и YOLOv11 совпадают.

C. *Сравнение YOLO и Faster R-CNN*

Для оценки различий между одностадийными и двухстадийными подходами было выполнено сравнение моделей YOLOv8, YOLOv11 и Faster R-CNN. В таблице 3 представлены итоговые показатели качества детектирования.

Таблица 3 — Сравнение моделей YOLO и Faster R-CNN

Модель	precision	recall	mAP50	mAP50-95
YOLOv 8	0.79	0.62	0.71	0.42
YOLOv 11	0.81	0.64	0.73	0.45
Faster R-CNN	0.84	0.68	0.77	0.54

Результаты показывают, что модель Faster R-CNN обеспечивает незначительно высокие значения по всем метрикам, что связано с двухстадийным механизмом детектирования и более точной локализацией объектов. Однако данная модель характеризуется значительно большим временем обработки по сравнению с архитектурами семейства YOLO, для сравнения YOLOv8 сделала те же вычисление в 2,23 раза быстрее, а YOLOv11 в 1,19 раз быстрее.

Графики метрик во время обучения у Faster R-CNN имеют более плавный рост чем графики моделей YOLO.

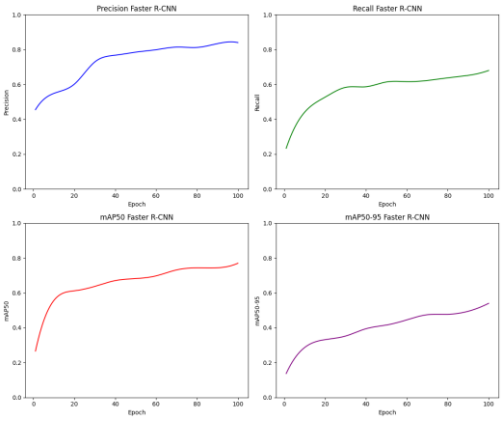


Рис. 11 — Графики метрик Faster R-CNN

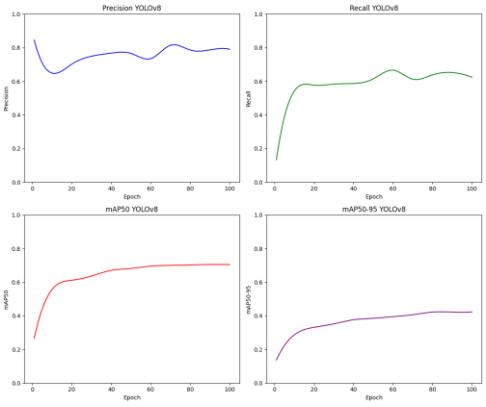


Рис. 12 — Графики метрик YOLOv8

D. *Анализ результатов*

На основе проведённого сравнительного анализа установлено, что архитектуры семейства YOLO обеспечивают оптимальное соотношение точности детектирования и вычислительной эффективности, что обусловливает их целесообразность для применения в прикладных системах анализа дорожной сцены и системах реального времени. В то же время двухстадийная архитектура Faster R-CNN демонстрирует более высокие показатели точности детектирования за счёт поэтапного формирования и уточнения областей интереса, однако характеризуется повышенной вычислительной сложностью, что ограничивает её использование задачами, в которых требования к быстродействию не являются критичными [8, 9].

V. Заключение

В рамках данной работы была рассмотрена задача детекции объектов дорожной сцены на цифровых изображениях с использованием методов компьютерного зрения и сверточных нейронных сетей. Актуальность исследования обусловлена высокой практической значимостью автоматического анализа дорожной обстановки для интеллектуальных транспортных систем, систем помощи водителю и автономных транспортных средств.

В ходе исследования был сформирован и размечен собственный набор данных изображений дорожной сцены с применением инструмента CVAT.AI, включающий основные классы объектов, характерные для городской дорожной инфраструктуры. Подготовленный датасет был разделен на обучающую, валидационную и тестовую выборки и опубликован в открытом доступе.

Для решения задачи детекции объектов были реализованы и обучены модели, относящиеся к различным архитектурным подходам, а именно одностадийные детекторы семейства YOLO и двухстадийная модель Faster R-CNN. Проведен сравнительный анализ качества работы моделей по показателям точности, полноты и средней точности при различных порогах перекрытия.

Полученные результаты показали, что модели семейства YOLO обеспечивают оптимальный баланс между точностью и скоростью обработки изображений, что делает их целесообразными для использования в системах реального времени. Модель Faster R-CNN продемонстрировала более высокие значения показателей качества детектирования, однако характеризуется значительно большей вычислительной сложностью и меньшей скоростью работы, что ограничивает ее применение в задачах с жесткими требованиями к быстродействию.

Таким образом, результаты исследования подтверждают эффективность применения методов глубокого обучения для задач детекции объектов дорожной сцены и демонстрируют целесообразность использования одностадийных архитектур в прикладных системах компьютерного зрения.

VI. Список литературы

[1] Зайцева, Е. В. Цифровые преобразования транспортной логистики в стратегии инновационного развитии предприятий цементной отрасли / Е. В. Зайцева // General question of world science : Collection of scientific papers on materials X International Scientific Conference, Amsterdam, 31 июля 2020 года. – Amsterdam: Наука России, 2020. – С. 36–38. – DOI 10.18411/gq-31-07-2020-08. – EDN KSKYIZ. https://elibrary.ru/item.asp?id=43321820

[2] Садеков, Р. На объектах транспортирования опасных веществ / Р. Садеков // ТехНадзор. – 2013. – № 11(84). – С. 44–45. – EDN WWSIQN. https://elibrary.ru/item.asp?id=27176595

[3] Леонов, И. Ю. Классификация транспортных средств компьютерным зрением / И. Ю. Леонов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей НТС, Москва, 30 декабря 2023 года. – Москва: НИТУ «МИСИС», 2023. – С. 46–52. – EDN JSRESQ. https://elibrary.ru/item.asp?id=65641260&pff=1

[4] Грищенко, Д. И. Распознавание людей на железнодорожной инфраструктуре / Д. И. Грищенко // Там же. – 2023. – С. 118–121. – EDN CQHKCE. https://elibrary.ru/item.asp?id=65641250&pff=1

[5] Фомина, А. А. Классификация драгоценных камней при помощи компьютерного зрения / А. А. Фомина // Там же. – 2023. – С. 28–33. – EDN SFXFCG. https://elibrary.ru/item.asp?id=65641230&pff=1

[6] Лим, В. Л. Исследование вопроса распознавания светофоров / В. Л. Лим // Там же. – 2023. – С. 34–39. – EDN KDXQCK. https://elibrary.ru/item.asp?id=65641258&pff=1

[7] Гужва, Н. С. Алгоритмы локализации и сопоставления светофоров в системах помощи водителя / Н. С. Гужва, Р. Н. Садеков // XXXI Санкт-Петербургская международная конференция по интегрированным навигационным системам. – СПб., 2024. – С. 167–173. – EDN DYPBIF. https://elibrary.ru/item.asp?id=72041388

[8] Абакумов, А. А. Вопросы сегментации дорожного слоя / А. А. Абакумов, В. О. Хуако // Там же. – 2023. – С. 40–45. – EDN UWTAKJ. https://elibrary.ru/item.asp?id=65641259&pff=1

[9] Солодов, С. В. Методы исследования дефектов дорожного покрытия в видеопотоке камер портативных устройств / С. В. Солодов, С. В. Проничкин // Новые материалы и перспективные технологии. – Москва, 2019. – Т. I. – С. 418–421. – EDN SZLIDC. https://elibrary.ru/item.asp?id=42521412

[10] Пазычев, Д. Б. Комплекс навигации для беспилотного летательного аппарата / Д. Б. Пазычев, К. С. Бакулев // XXXI Санкт-Петербургская международная конференция по интегрированным навигационным системам. – СПб., 2024. – С. 200–207. – EDN PMNQMR. https://elibrary.ru/item.asp?id=72041446

[11] Сравнение различных методов нахождения эффекта масштаба в нерадиальных моделях анализа среды функционирования / В. Е. Кривоножко, А. П. Афанасьев, Ф. Р. Форсунд, А. В. Лычев // Автоматика и телемеханика. – 2022. – № 7. – С. 152-168. – DOI 10.31857/S0005231022070091. – EDN AFMFXJ. https://elibrary.ru/item.asp?id=48592416

ГРНТИ 28.23.15
ASJC 1707
OECD 1.2

Детекция элементов пользовательского интерфейса на скриншотах веб приложений для автоматизированного контроля изменений

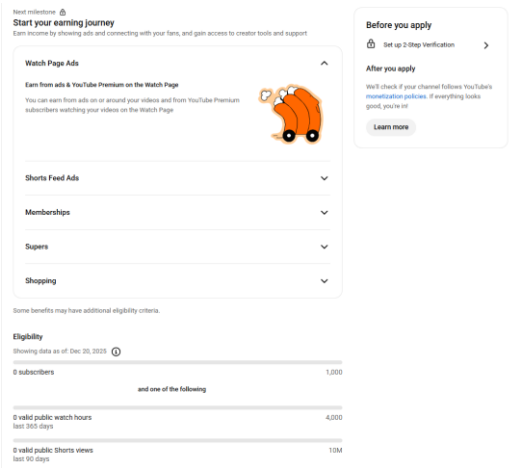
Н. А. Исмаилова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2515775@edu.misis.ru

***Аннотация* — в работе рассматривается задача обнаружения элементов пользовательского интерфейса (UI) на скриншотах веб-страниц. Цель - построить воспроизводимый пайплайн: от разметки данных и конвертации аннотаций до обучения модели детекции и последующего применения результатов для контроля изменений интерфейса. Показано, что объединение детекции и структурной постобработки позволяет автоматизировать QA-анализ визуальных изменений интерфейса. В статье приведены рекомендации по воспроизводимости, метрики качества и примеры визуализации.**

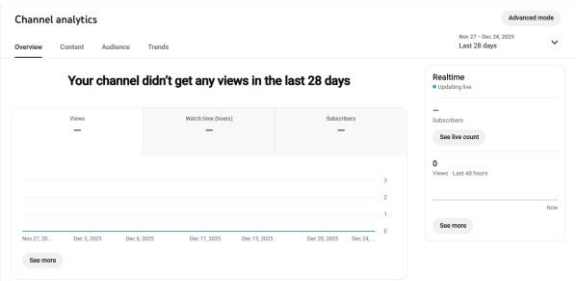
Ключевые слова — визуальная регрессия, детекция объектов, компьютерное зрение, контроль изменений, пользовательский интерфейс

I. Введение

Веб интерфейс можно рассматривать как сложную визуальную сцену, состоящую из множества компонентов, которые образуют иерархию и задают пользовательские сценарии. На практике поддержка качества интерфейса часто упирается в визуальную регрессию: после изменений в коде могут сместиться элементы, исчезнуть кнопка, “поехать” навигация, нарушиться визуальный приоритет блоков. Проверка таких ошибок вручную плохо масштабируется и сильно зависит от внимательности проверяющего. Примеры типовых интерфейсов веб-приложений приведены на рис. 1.



a)



b)

Рис. 1. Примеры исходных скриншотов веб-интерфейса: (a) интерфейс с высокой плотностью элементов; (b) интерфейс с иерархической навигацией.

Задача данной работы заключается в исследовании возможности **автоматической детекции элементов пользовательского интерфейса на скриншотах веб приложений** как базового шага для последующего контроля изменений. Если модель стабильно выделяет элементы (кнопки, поля ввода, иконки, текстовые блоки и т.д.), дальше можно формализовать проверку: сравнивать присутствие элементов, их относительные размеры и расположение, а также изменения структуры страницы между версиями.

Вклад работы:

1. сформирован и опубликован пилотный набор данных UI-скриншотов в форматах COCO и YOLO

2. реализован воспроизводимый пайплайн подготовки данных и обучения модели детекции

3. Используются три детектора с разным балансом скорости и точности:

- **YOLOv8n** - компактная и быстрая модель для массового инференса;
- **YOLOv8s** - компромисс между скоростью и качеством;
- **Faster R-CNN** - более точная модель для детального анализа.

Практическая ценность предлагаемого подхода связана с задачами автоматизированного тестирования и мониторинга пользовательских интерфейсов. Детекция UI-элементов позволяет перейти от визуального сравнения скриншотов к воспроизводимому контролю

изменений и использованию количественных метрик качества. В рамках работы формируется собственный набор данных, выполняется разметка классов интерфейсных компонентов и создаётся основа для обучения и оценки моделей детекции.

II. Наборы данных

A. *Требования к датасету для UI детекции*

Данные для детекции элементов интерфейса должны отражать реальную вариативность веб страниц: разные сетки и компоновки, плотность контента, размеры шрифтов, темы оформления, наличие изображений и иконок. Критически важны два фактора:

1. **мелкие объекты** (иконки, чекбоксы, элементы в навигации), которые занимают малую долю площади кадра и часто дают основные ошибки детекции
2. **контекст и близкое соседство** (текст рядом с кнопкой, иконка внутри кнопки, элементы внутри карточек), из-за чего даже корректная рамка может быть неоднозначной

Это напрямую влияет на требования к разметке и на то, какие архитектуры имеет смысл сравнивать, что согласуется с общими выводами о роли постановки задачи и данных в работах по детекции [1], [2]. В прикладных задачах компьютерного зрения качество данных и условия съемки оказывают существенное влияние на стабильность детекции. [3].

B. *Собственный набор данных скриншотов веб приложений*

В рамках данной работы был сформирован пилотный набор данных скриншотов веб-приложений, ориентированный на задачи детекции элементов пользовательского интерфейса. Набор данных включает интерфейсы с различной структурой, плотностью элементов и визуальной иерархией, что позволяет оценить поведение моделей детекции в условиях, характерных для UI-сценариев.

UI-данные отличаются от природных изображений: объекты часто мелкие, плотные, выровнены по сетке и имеют близкую семантику. Поэтому качество разметки (точность рамок, единообразие правил) напрямую влияет на метрики и переносимость модели. [1], [2]. Основные количественные характеристики набора данных приведены в табл. 1.

На текущем этапе он включает:

Таблица 1
Статистика датасета.

Параметр	**Значение**
Число изображений	10
Число аннотаций	624
Среднее число объектов на изображение	62.4
Диапазон размеров изображений	1291x681 - 1916x1057
Train/Val split	80/20 (seed 42)

Доступность данных. Датасет опубликован по адресу: https://huggingface.co/datasets/nargizDev/ui-elemets-detection-coco. [4] Данные представлены скриншотами интерфейсов и аннотациями в формате COCO, экспортированными из CVAT.

Фиксация скриншотов выполнялась в воспроизводимых условиях отображения (единый подход к масштабу и разрешению, отсутствие “случайных” артефактов интерфейса). Такой контроль условий важен, чтобы изменения в разметке и в результатах детекции отражали именно структуру интерфейса, а не шум от разных параметров съемки.

C. *Классы объектов и статистика разметки*

В пилотной версии набора данных используются 9 классов UI-компонентов:

- **button** (кнопка)
- **text** (текстовый блок)
- **icon** (иконка)
- **image** (изображение/медиа)
- **input** (поле ввода)
- **dropdown** (выпадающий список)
- **checkbox** (чекбокс)
- **navbar** (навигационная панель)
- **card** (карточка/контейнер)

Распределение размеченных объектов по классам для пилотного набора данных представлено на рис. 2.

- text: 256
- icon: 225
- button: 77
- image: 33
- navbar: 11
- card: 11
- dropdown: 5
- input: 5
- checkbox: 1

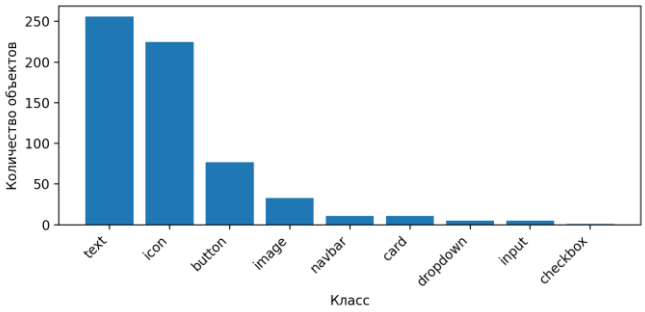


Рис. 2. Распределение размеченных объектов по классам (пилотный набор данных)

Пайплайн включает: разметка в CVAT > COCO > подготовка датасета > обучение детекторов > инференс > постобработка > сравнение интерфейсов. Общая схема используемого пайплайна представлена на рис. 3.

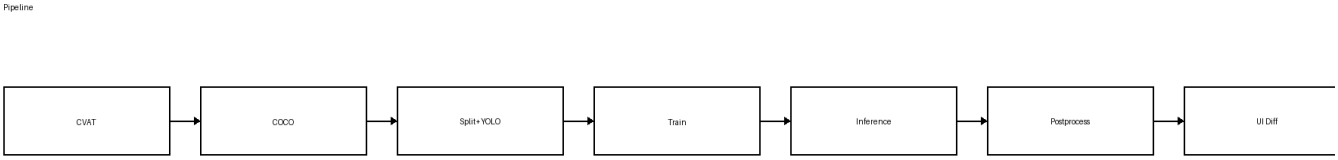


Рис. 3. Общая схема пайплайна подготовки данных и обучения моделей детекции.

Набор имеет выраженную особенность, характерную для UI: высокая доля мелких объектов. В частности, класс **icon** преимущественно представлен очень маленькими областями, что делает задачу более чувствительной к качеству разметки и выбору модели.

D. Инструмент разметки и формат хранения

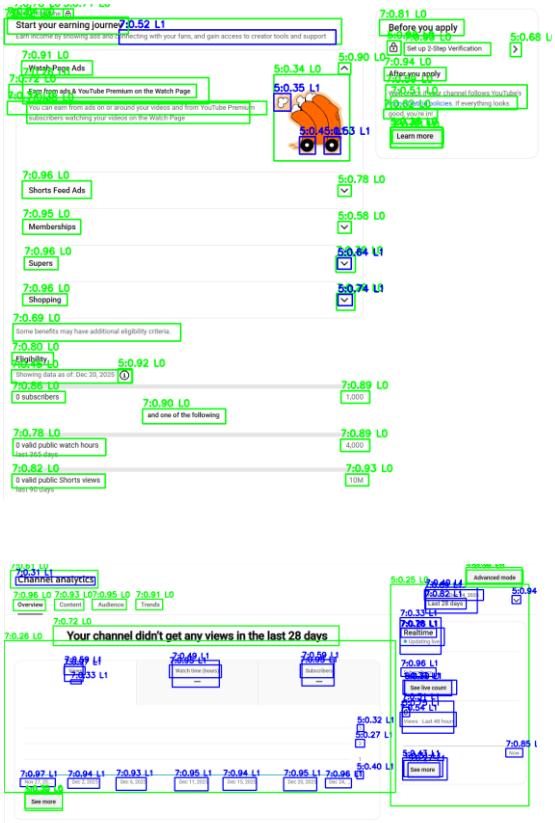
Разметка была выполнена с использованием инструмента **CVAT** в рамках задачи детекции объектов, где для каждого объекта определены ограничивающие рамки (bounding boxes). Подходы к детекции объектов с использованием ограничивающих рамок и анализу ошибок локализации рассматривались, в том числе, в работах по детекции документов в реальном времени. [6]. Данные были экспортированы в формате COCO JSON, который широко используется в задачах детекции объектов и обеспечивает удобное хранение аннотаций и метаданных. [7]. Формат COCO предоставляет возможность хранения списка изображений, классификационных категорий и аннотаций, а также допускает последующее расширение набора данных без необходимости изменения структуры данных.

III. Методы и нейросетевые архитектуры

A. Постановка задачи

Задача формулируется как детекция объектов на изображении: по входному скриншоту требуется предсказать набор ограничивающих рамок и класс каждого найденного элемента интерфейса. Такой формат удобен тем, что результат детекции можно напрямую использовать как структуру страницы: сравнивать наличие элементов, их относительное положение, размеры и взаимное расположение блоков между версиями интерфейса.

Примеры визуализации разметки UI-элементов на скриншотах веб-приложений приведены на рис. 4. Они иллюстрируют высокую плотность объектов, наличие мелких элементов и вложенность компонентов, характерные для UI-сценариев.



a)

b)

Рис. 4. Пример визуализации разметки для скриншотов, приведенных на рис. 1: а) разметка к рис. 1а; б) разметка к рис. 1б

С практической точки зрения детекция UI отличается от “обычных” объектов тем, что:

- на одном изображении объектов много (плотная сцена),
- существенная доля объектов мелкие (иконки, чекбоксы),
- много повторяющихся паттернов (списки, карточки, меню),
- встречаются частичные перекрытия и вложенность (иконка внутри кнопки, текст внутри карточки).

В связи с этим выбор архитектуры и протокола обучения должен определяться особенностями данных, а не стремлением использовать наиболее сложную модель. Общие ориентиры по современным детекторам и их свойствам удобно брать из обзорной работы [1], а также из прикладного анализа постановки детекции и свойств моделей. [2].

Анализ архитектур нейронных сетей и их применимости к задачам распознавания изображений и детекции объектов представлен, в том числе, в работах. [8].

B. *Одностадийные детекторы*

Одностадийные детекторы (семейство YOLO и близкие подходы) выполняют предсказание рамок и классов за один проход, что обычно дает хорошее соотношение скорость/точность. Для прикладных задач, где требуется быстро получать результаты и масштабироваться на большое число скриншотов, этот класс моделей логично рассматривать как базовый.

Для UI это полезно еще и потому, что типовой пайплайн проще: меньше компонентов, быстрее экспериментировать с гиперпараметрами, проще воспроизводимость. При этом слабое место одностадийных детекторов в UI-сценариях часто связано с маленькими объектами и плотным соседством, поэтому дальнейшие эксперименты должны явно проверять качество по классам icon, checkbox, dropdown, input.

C. *Двухстадийные детекторы*

Двухстадийные модели (например, Faster R-CNN) сначала формируют кандидаты областей, затем уточняют класс и рамку. Их плюс в том, что они часто лучше держат качество на сложных и мелких объектах, особенно когда сцена перегружена. Минус, как правило, более высокая вычислительная стоимость и более тяжелая настройка.

Faster R-CNN приведён в обзорной части как реперная архитектура для качественного сопоставления семейств детекторов; экспериментальная часть сосредоточена на YOLOv8 в связи с пилотным характером датасета. Обзор и сравнение семейств детекторов приведены в [1], [2].

D. *Трансформерные детекторы*

Трансформерные детекторы (в том числе RT-DETR и близкие варианты) используют механизмы внимания и часто лучше учитывают глобальный контекст изображения. В UI это потенциально полезно, потому что элементы интерфейса “живут” в композиции: меню, карточки, контентные зоны связаны общей структурой.

С другой стороны, трансформерные подходы могут быть более чувствительны к объему данных и стабильности разметки. Поэтому их включение в сравнение имеет смысл после того, как базовый датасет станет больше и появится более устойчивая статистика по метрикам.

E. *Обоснование выбора моделей для эксперимента*

В рамках исследования предлагается следующий минимально достаточный и репрезентативный набор для сопоставления различных архитектурных подходов:

- базовая модель: одностадийный детектор (класс YOLO),
- альтернативная архитектура: Faster R-CNN как двухстадийный детектор,
- перспективная альтернатива: RT-DETR как трансформерный детектор.

Такой набор архитектур отражает разные принципы построения детекторов и задает контекст для дальнейшего сравнения. В экспериментальной части данной работы оценивается YOLOv8; Faster R-CNN и RT-DETR рассматриваются как альтернативы для последующего сравнения после расширения датасета.

IV. Подготовка данных и протокол эксперимента

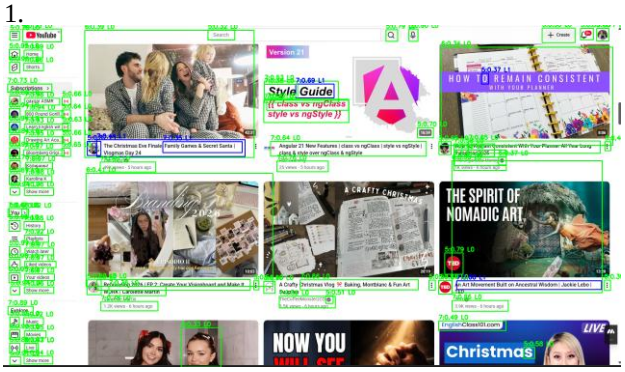
A. *Разметка и контроль качества*

Разметка выполняется в формате ограничивающих рамок. Для задач UI важно ввести единые правила, иначе модель будет учиться на противоречиях. В рамках работы был сформирован минимально достаточный набор правил разметки:

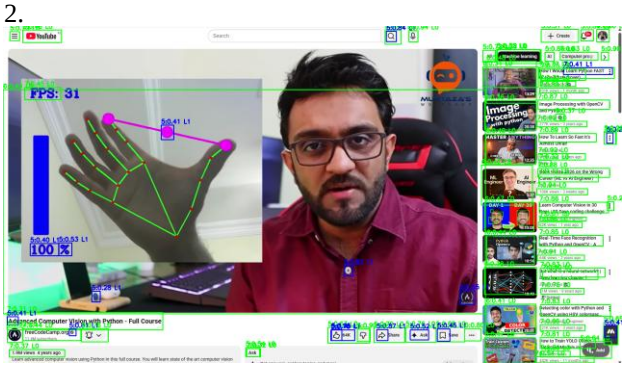
1. Рамка должна покрывать видимую область элемента без чрезмерного захвата “пустоты”.
2. Если элемент вложен в другой (иконка внутри кнопки), размечаются оба, но рамки не должны быть произвольными.
3. Текст размечается как блок (text), а не посимвольно, чтобы не создавать искусственную дробность.
4. Для контейнерных элементов (navbar, card) применяется правило разметки как целостных структур, без попытки “обвести всю страницу”.

Примеры разметки UI-компонентов приведены на рис. 5.

1.



2.



3.

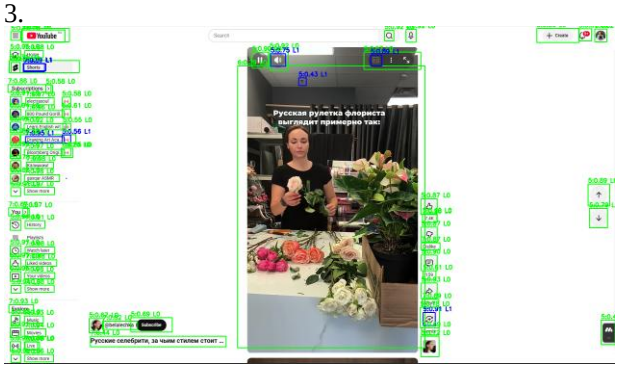


Рис. 5. Пример разметки UI-компонентов ограничивающими рамками (COCO)

Поскольку в пилотном наборе много мелких объектов (особенно icon), отдельно фиксируется проблема смещения рамок на несколько пикселей: для маленьких объектов это резко меняет IoU и итоговые метрики. Данный эффект учитывался при интерпретации результатов эксперимента в главе V.

B. Разделение на обучающую и тестовую выборки

В силу малого объема данных используется hold-out разбиение с фиксированным seed, ориентированное на воспроизводимость, а не на статистическую устойчивость метрик. Для корректного описания экспериментальной методики целесообразно выделить два режима эксперимента. [1], [7].

Режим 1 (пилотный). Hold-out разбиение с фиксированным seed и дополнительными ограничениями:

- разбиение выполняется так, чтобы все классы, присутствующие в наборе, встречались и в обучающей, и в тестовой части (минимальная стратификация по классам) [7].
- исключается “утечка” визуально близких данных: если есть серии однотипных скриншотов одного экрана (разные состояния одной страницы), они должны попадать только в одну из частей (групповое разбиение по источнику скриншотов) [8].
- параметры разбиения фиксируются (seed, список файлов в train и test), чтобы обеспечить воспроизводимость эксперимента. [1].

Режим 2 (основной, после расширения датасета). Стандартный протокол оценки для детекции:

- разбиение train/val/test по изображениям (например, 70/15/15 или 80/10/10) со стремлением к балансировке классов [6].
- альтернативно k-fold по изображениям при умеренном объеме данных, чтобы снизить зависимость оценки от конкретного разбиения. [1].
- контроль отсутствия дубликатов и near-duplicate между частями, что особенно важно для домена UI из-за повторяющихся шаблонов. [6].

C. Баланс классов

По текущей статистике классы распределены неравномерно: доминируют text и icon, редкие классы checkbox, dropdown, input представлены единично. Для задач детекции дисбаланс приводит к деградации качества по редким классам даже при приемлемом среднем mAP, поэтому меры контроля дисбаланса должны быть явно зафиксированы в протоколе. [2], [6].

Предлагаемые меры:

- учет метрик не только в среднем виде (mAP), но и по классам (AP per class) с отдельной фиксацией качества на редких категориях; [1].
- целевое пополнение редких классов при следующем сборе данных, с минимальными квотами на каждый класс; [6].
- опционально объединение крайне редких классов на стадии пилотного эксперимента в укрупненную категорию, если иначе невозможно получить устойчивые оценки (решение фиксируется в описании разметки); [7].
- применение приемов обучения, устойчивых к дисбалансу: взвешивание функции потерь, oversampling изображений с редкими объектами, варианты focal loss, если они поддержаны реализацией; [1], [2].
- приоритетная проверка качества разметки для редких классов, так как единичные ошибки сильно искажают оценку; [7].

D. Аугментации и синтетические данные

Для скриншотов интерфейсов агрессивные фотометрические аугментации часто ухудшают сигнал обучения, так как UI содержит мелкие элементы и текст, чувствительные к искажениям. Поэтому применяемые аугментации должны моделировать реалистичные вариации отображения и процесса получения скриншота, не разрушая геометрию и читабельность. [2], [8].

Рекомендуемые аугментации:

- масштабирование и ресайз (multi-scale), так как элементы встречаются при разных разрешениях [2].
- небольшие сдвиги и кропы, имитирующие вариативность viewport, с корректной трансформацией bounding boxes [2].
- артефакты сжатия (JPEG-compression) и слабое размытие, имитирующие пересохранение и передачу изображений [8].
- слабые изменения яркости и контраста в узких пределах, если в датасете присутствуют разные темы и режимы отображения [8].

Аугментации, применяемые осторожно или исключаемые:

- сильные повороты и перспективные искажения, как правило, нерелевантны для UI [8].
- сильные цветовые искажения, ухудшающие распознавание текста и границ [8].

Синтетические данные. Для увеличения разнообразия и покрытия редких классов допускается использование синтетических изображений при контролиру-

емой генерации и валидации качества. Это может включать рендеринг типовых экранов с вариациями темы, языка, размера окна, а также генерацию состояний компонентов (hover, focus, disabled, error). Подходы к использованию синтетики в задачах детекции и сегментации рассматриваются как один из способов управляемого расширения обучающей выборки. [7].

V. Анализ результатов и обсуждение

В этой главе фиксируются метрики, способ интерпретации качества, а также типовые ошибки детектора на скриншотах веб приложений. Отдельный акцент сделан на том, что пилотный набор мал по объему, поэтому численные оценки следует трактовать как предварительные и обязательно сопровождать анализом ошибок и качественными примерами.

A. *Метрики качества и протокол оценки*

Для задачи детекции элементов интерфейса используются стандартные метрики объектного детектирования, которые позволяют раздельно оценивать точность локализации и корректность классификации. В качестве основных метрик приняты:

1. Precision (точность). Доля корректных предсказаний среди всех предсказанных объектов. Более высокое значение Precision соответствует меньшей доле ложных срабатываний (False Positive).
2. Recall (полнота). Доля найденных объектов среди всех объектов разметки. Рост Recall означает снижение числа пропусков (False Negative).
3. F1 score. Сводная метрика, балансирующая Precision и Recall, полезна при сравнении конфигураций при фиксированном пороге confidence.
4. mAP (mean Average Precision). Основная метрика для детекторов, учитывает ранжирование предсказаний по уверенности и качество пересечения рамок через IoU.

Итоговая оценка качества включает две плоскости: (1) точность детекции и (2) производительность. Иными словами, это два блока параметров: качество и скорость. Для точности используются mAP@0.5 и mAP@0.5:0.95, а также precision/recall и F1 по классам. Для производительности измеряется время инференса на изображение (мс) и FPS. Дополнительно строятся матрицы ошибок по классам при IoU >= 0.5 (однозначное сопоставление предсказаний и GT).
В экспериментах использованы пороги: conf=0.1 и IoU=0.5.

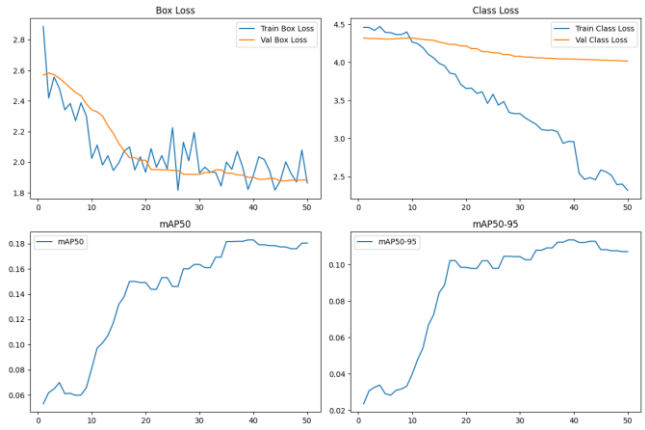


Рис. 6. Кривые обучения и метрики детектора (loss/precision/recall/mAP)

Для корректности воспроизводимости протокол оценки фиксируется явно:

- порог IoU для зачета совпадения предсказания с разметкой (например 0.5)
- стратегия выбора confidence threshold (либо фиксированный порог, либо анализ кривых precision-recall)
- способ агрегации по классам, включая per-class AP, поскольку для UI типичны редкие и визуально похожие классы

Выбор набора метрик и протокола оценки ориентирован на воспроизводимость и сопоставимость результатов на пилотном датасете. Обзор подходов к оценке детекторов приведен в [1], [2].

B. *Количественные результаты на тестовой выборке*

Если сравниваются режимы (пилотный и основной после расширения датасета), то таблица дублируется для каждого режима или добавляются столбцы “Режим 1” и “Режим 2”.

Важно отдельно пояснить, почему на пилотном наборе значения могут быть нестабильны: малый объем данных дает высокую дисперсию оценки, а любое смещение в сторону одного типа экранов или одной группы компонентов резко меняет итоговую метрику. В работах, посвященных формированию обучающих выборок, эта проблема обычно выделяется отдельно, поскольку именно дизайн набора данных определяет достоверность последующей оценки. В контексте данной модели это напрямую связано с решениями по разбиению и балансировке, описанными ранее, и соотносится с выводами о датасетах в [7]. В данной работе количественные значения приведены для иллюстрации различий между архитектурами и не претендуют на статистически устойчивую оценку качества

C. *Качественный анализ и типовые ошибки детектора*

Даже при наличии метрик, для практической задачи контроля изменений интерфейса важны ошибки, которые приводят к неверным выводам в QA сценари-

ях. Поэтому результаты следует сопровождать “разбором провалов” на реальных скриншотах.

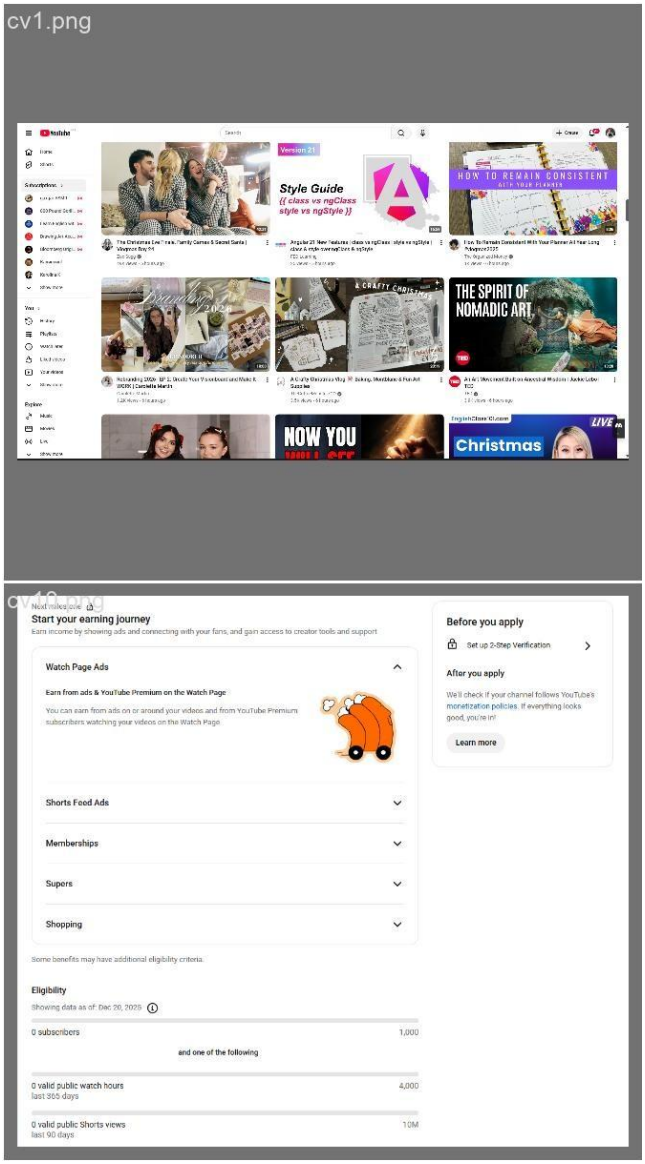


Рис. 6. Пример предсказаний детектора на валидационном изображении (bounding boxes)

Пример предсказаний детектора на валидационном изображении с визуализацией ограничивающих рамок приведён на рис. 6.

1. Ложные срабатывания (False Positive). На UI чаще всего возникают в двух ситуациях:
   - мелкие декоративные элементы или фрагменты текста детектируются как полноценный компонент
   - повторяющиеся паттерны верстки (например иконки, разделители, тени, границы) воспринимаются моделью как объекты целевого класса

На практике FP особенно опасны для сценариев мониторинга, потому что дают шум и усложняют автоматическое сравнение версий интерфейса. Примеры ложных срабатываний и пропусков объектов на валидационных изображениях приведены на рис. 10.

2. Пропуски объектов (False Negative). Частые причины:
   - малый размер элемента на скриншоте (иконки, чекбоксы, маленькие кнопки)
   - перекрытия (dropdown поверх контента, модалки, всплывающие подсказки)
   - низкий контраст и нестандартные темы (dark mode, кастомные палитры)

Для UI пропуски критичнее, чем FP, если задача сформулирована как “не пропустить регрессию” или “не потерять ключевые элементы”. Поэтому в обсуждении важно отдельно проговорить, какой компромисс принят между Precision и Recall.

3. Ошибки локализации. Даже при верном классе рамка может быть неточной: захватывать лишнюю “пустоту” или, наоборот, обрезать часть элемента. Для последующего анализа визуальной иерархии это принципиально, потому что геометрия объектов является входом для вычисления отношений “выше-ниже”, “внутри”, “рядом” и т. п.

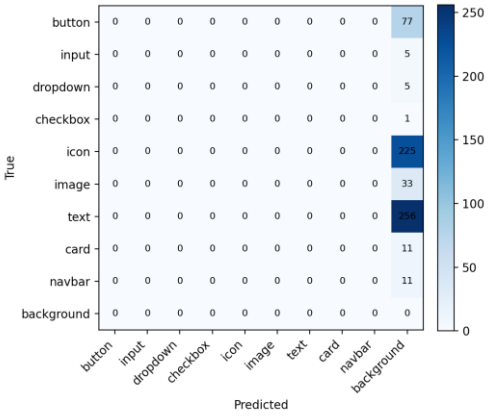
Типовая причина ошибок локализации на интерфейсах это мелкие объекты и плотная компоновка. В обзорах по архитектурам детекции отмечается, что разные семейства моделей по-разному чувствительны к размерам объектов и к особенностям текстур, что полезно учитывать при выборе базовой архитектуры и разрешения входного изображения. [1], [2].

4. Путаница между классами UI классы часто семантически близки и визуально пересекаются: button vs text, input vs dropdown, icon vs checkbox. Наиболее частые случаи путаницы классов на валидационной выборке приведены в табл. 1. Если такие пары классов есть в разметке, то стоит явно указать самые проблемные пары и показать примеры.

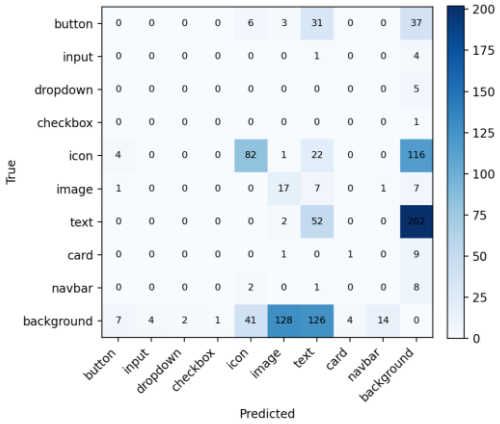
Истинный класс	Предсказан	Комментарий
button	text	текст рядом с кнопкой
input	dropdown	визуально похожие поля
icon	button	иконка внутри кнопки
checkbox	button	маленькие объекты
navbar	card	пересечения областей

Таблица 1. Топ-5 наиболее частых путаниц классов (валидация)

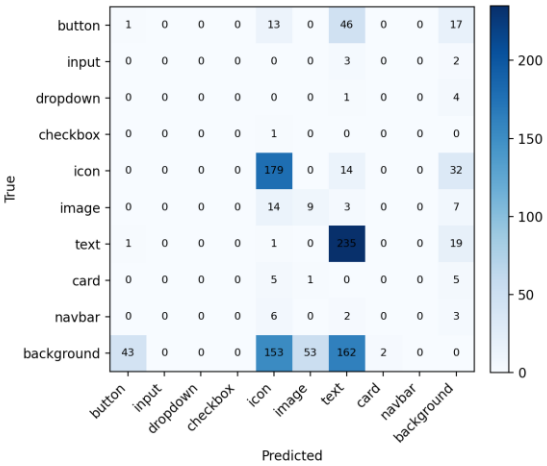
Это особенно важно при несбалансированных классах, потому что средний mAP может выглядеть приемлемо, но редкие классы будут провалены. Для детального анализа ошибок по классам построены матрицы ошибок при IoU ≥ 0.5 для всех рассматриваемых детекторов (рис. 7).



a)



b)



c)

Рис. 7. Матрицы ошибок для трех детекторов (IoU >= 0.5): a) YOLOv8n, b) YOLOv8s, (c) Faster R-CNN.

Значения приведены для одного фиксированного разбиения и используются для сравнительного анализа архитектур.

Модель	mAP@0.5	mAP@0.5:0.95	Precision	Recall	F1
YOLOv8n	0.0000	0.0000	0.0000	0.0000	0.0000
YOLOv8s	0.0755	0.0406	0.1460	0.1487	0.1203
Faster R-CNN	0.2969	0.1522	0.3286	0.3313	0.2779

Таблица 2. Сравнение детекторов по точности (mAP@0.5, mAP@0.5:0.95; precision/recall/F1 macro).

Модель	FPS	Время на изображение, мс
YOLOv8n	1.11	900.54
YOLOv8s	0.95	1057.48
Faster R-CNN	0.20	4916.62

Таблица 3. Сравнение детекторов по производительности.

Время инференса приведено для одного изображения на CPU (Intel Core Ultra 7 155H), без batch-обработки.

D. *Влияние датасета, аугментаций и синтетических данных*

Для интерфейсов агрессивные фотометрические аугментации действительно могут ухудшать качество: интерфейс обычно “чистый”, резкий и с контролируемыми цветами, поэтому сильные blur, noise, color jitter могут вносить дополнительный шум в обучение модели. В то же время умеренные трансформации, которые имитируют реальные условия получения скриншота, могут быть полезны:

- небольшое масштабирование и ресайз
- легкие изменения яркости и контраста в узких пределах
- незначительные сдвиги и кроп, если это похоже на реальный capture

Отдельный блок это синтетические данные. В задачах, где трудно быстро расширить разметку, синтетическая генерация может быть способом увеличить разнообразие и покрытие редких классов. Но ее нужно описывать аккуратно: что именно генерировалось, насколько синтетика похожа на реальные скриншоты, какие классы ей усиливались, как проверялось, что модель не “переучилась на синтетику”. Общие подходы к построению выборок и использованию синтетических изображений для сегментации и детекции можно найти в [7].

В рамках этой статьи корректно сформулировать так: синтетические данные рассматриваются как инструмент увеличения вариативности и балансировки, а не как замена реальных примеров.

E. *Практическая интерпретация для задачи контроля изменений интерфейса*

Полученные метрики и выявленные типы ошибок важно интерпретировать с точки зрения прикладной задачи контроля изменений интерфейса. Для QA сценариев важен не только факт детекции, но и стабильность структуры:

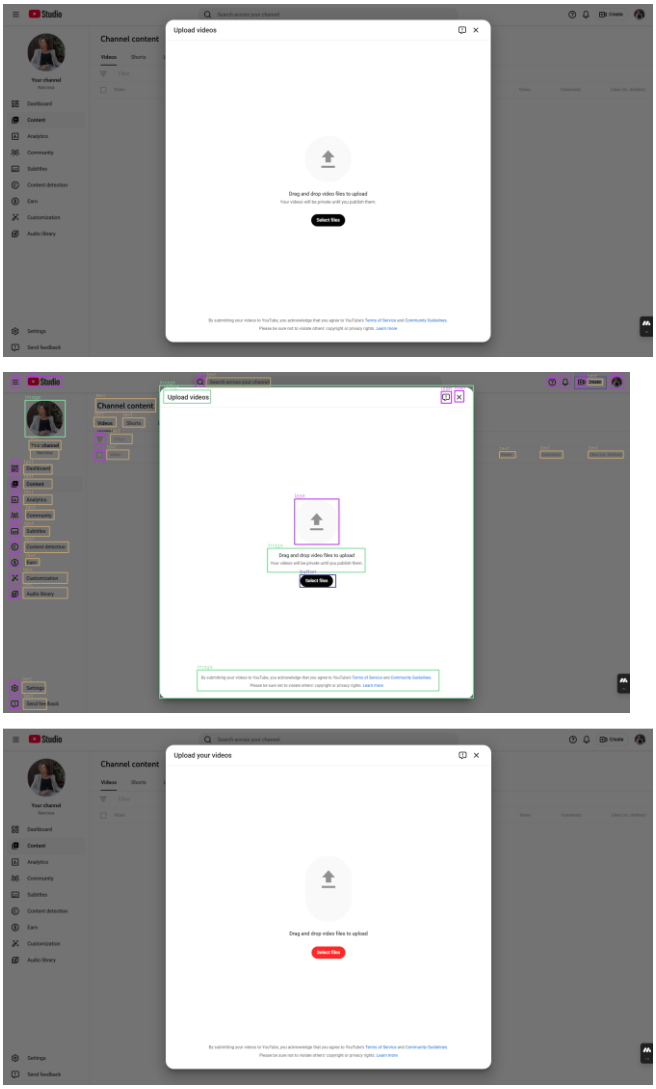
- детектор должен находить ключевые элементы устойчиво на разных страницах
- ошибки локализации не должны разрушать вычисление относительных отношений между элементами
- рост FP не должен превращать сравнение версий в “шумовую кашу”

Поэтому итоговая оценка качества должна включать две плоскости:

1. классические метрики детекции (mAP, Precision, Recall)
2. прикладную проверку на кейсах “до/после”, когда сравниваются скриншоты двух версий и проверяется, корректно ли алгоритм выделяет значимые изменения (например исчезновение кнопки, смещение блока, изменение структуры формы)

Также корректно указать ограничения. В задачах CV, связанных с анализом изображений в “грязных” условиях и при высокой вариативности источника, это считается нормальным подходом: сначала воспроизводимый протокол, затем масштабирование данных и повтор оценки. Аналогичная логика обсуждается и в работах, где анализируют CV методы на сложных и шумных данных. [8].

Пример выявления изменений интерфейса при сравнении двух версий скриншота приведён на рис. 8.



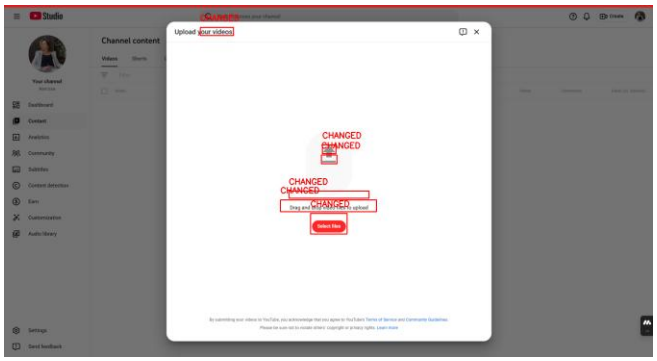


Рис. 8. Пример выявления изменений интерфейса при сравнении двух версий скриншота (пометки CHANGED)

Для анализа структурных изменений используется наложение предсказаний baseline и current версий интерфейса (рис. 9).

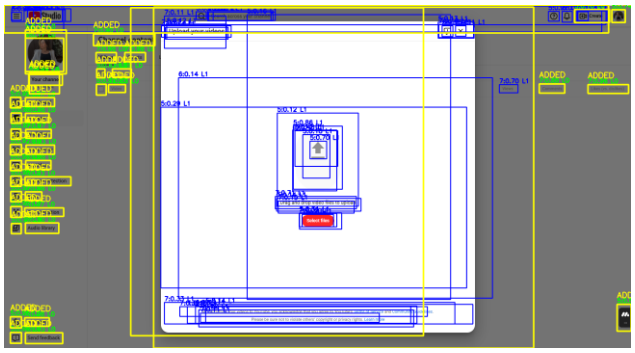


Рис. 9. Пример сравнения baseline/current с подсветкой изменений

На выбранном пороге conf=0.1 модель YOLOv8n не дала детекций, что привело к нулевым метрикам; для малых датасетов это ожидаемо.

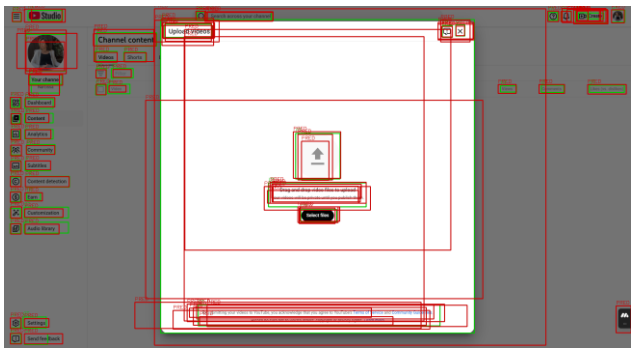


Рис. 10. Примеры ошибок детекции (false positive/false negative)

VI. Заключение

В работе рассмотрена задача детекции элементов пользовательского интерфейса на скриншотах веб приложений как базовый шаг для автоматизации визуального тестирования и контроля изменений интерфейса. Показано, что постановка задачи в виде детекции объектов с ограничивающими рамками позволяет перейти от субъективной ручной проверки к воспроизводимому анализу структуры страницы, основанному на координатах и типах компонентов.

Сформирован пилотный набор данных скриншотов веб интерфейсов и выполнена разметка классов интерфейсных элементов в формате bounding boxes. Проведена систематизация требований к данным и предложен протокол подготовки выборок, учитывающий риски утечки схожих скриншотов между частями набора и влияние дисбаланса классов на устойчивость оценки. Отдельно отмечено, что для UI домена критичны мелкие объекты и плотная компоновка, что повышает требования к качеству разметки и к выбору архитектуры детектора.

С методологической точки зрения работа опирается на современные подходы компьютерного зрения и детекции объектов, изложенные в обзорных и прикладных исследованиях. В качестве основы эксперимента определены семейства моделей, которые представляют разные архитектурные принципы (одностадийные, двухстадийные и трансформерные детекторы), что позволяет корректно сравнивать решения без подмены исследования сравнением версий одной модели.

Основные ограничения текущего этапа связаны с малым объемом пилотного датасета и неравномерным распределением объектов по классам, что влияет на стабильность численных метрик и требует дальнейшего расширения данных, особенно для редких категорий. При масштабировании экспериментов и увеличении объема данных важную роль играют вычислительные ресурсы и распределенные подходы обучения. [9]. В качестве направлений дальнейшей работы предлагается увеличение объема набора данных с контролем вариативности интерфейсов, балансировка классов, а также исследование влияния аугментаций и синтетического расширения выборки на качество детекции. Дополнительно планируется интеграция полученных результатов в практический сценарий визуальной регрессии, где детекция элементов используется для автоматического выявления значимых изменений между версиями интерфейса.

Итогом работы является воспроизводимая методика подготовки данных и постановки эксперимента для обучения моделей детекции элементов интерфейса, которая может быть использована при разработке систем автоматизированного тестирования и мониторинга качества пользовательских интерфейсов.

Литература

[1] Корчагин, В. Д. Анализ современных SOTA-архитектур искусственных нейронных сетей для решения задач классификации изображений и детекции объектов / В. Д. Корчагин // Программные системы и вычислительные методы. – 2023. – № 4. – С. 73-87. – DOI 10.7256/2454-0714.2023.4.69306. – EDN MZLZMK.

[2] Паластрова, В. Ю. Особенности модели нейронной сети для детекции объектов на изображении / В. Ю. Паластрова // Проблемы и перспективы развития АПК региона : Материалы Межвузовской научно-практической конференции, Пермь, 30 ноября 2023 года. – Пермь: ИПЦ Прокрость, 2023. – С. 143-148. – EDN HEITSZ.

[3] Методы распознавания и обработки изображений в процессе строительства нефтяных и газовых скважин / С. А. Усилин, В. В. Арлазаров, Д. Н. Путинцев, И. А. Тарханов // Информационные технологии и вычислительные системы. – 2020. – № 1. – С. 12-24. – DOI 10.14357/20718632200102. – EDN QKZACI.

[4] CVAT - Computer Vision Annotation Tool. Available at: https://www.cvat.ai/.

[5] Нечетов, Г. В. Анализ структур многослойных нейронных сетей для решения задачи распознавания изображений / Г. В.

Нечетов, А. И. Глущенко, А. Н. Скаковская // Ломоносов-2022 : Материалы XXIX Международной научной конференции студентов, аспирантов и молодых ученых, Севастополь, 14–22 апреля 2022 года. – Севастополь: Филиал МГУ в г. Севастополе, 2022. – С. 75-76. – EDN ZSFGOC.

[6] UI Elements Detection COCO Dataset. URL: https://huggingface.co/datasets/nargizDev/ui-elemets-detection-coco/tree/main

[7] Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C. L. Microsoft COCO: Common Objects in Context // Computer Vision - ECCV 2014. Lecture Notes in Computer Science. 2014. Vol. 8693. P. 740-755. DOI 10.1007/978-3-319-10602-1_48.

[8] Real time rectangular document detection on mobile devices / N. Skoryukina, D. P. Nikolaev, A. Sheshkus, D. Polevoy // Proceedings of SPIE - The International Society for Optical Engineering : 7, Machine Vision, Milan, 19–21 ноября 2014 года. Vol. 9445. – Milan, 2015. – P. 94452A. – DOI 10.1117/12.2181377. – EDN UFLWEH.

[9] Орлова, И. М. Создание обучающей выборки для сегментации и детекции объектов с помощью генерации синтетических изображений методами нейронных сетей / И. М. Орлова // Гагаринские чтения 2024 : Сборник тезисов докладов 50-ой Международной молодежной научной конференции, Москва, 09–12 апреля 2024 года. – Москва: ООО "Издательство "Перо", 2024. – С. 188-189. – EDN AGBSMI.

[10] Курочкин, И. И. Грид-система из персональных устройств на платформе BOINC для решения задач глубокого обучения / И. И. Курочкин, А. И. Прун // Оптико-электронные приборы и устройства в системах распознавания образов и обработки изображений : сборник материалов XVII Международной научно-технической конференции, Курск, 12–15 сентября 2023 года. – Курск: Юго-Западный государственный университет, 2023. – С. 252-254. – EDN XSQQLW.

[11] Ren S. Faster r-cnn: Towards real-time object detection with region proposal networks //arXiv preprint arXiv:1506.01497. – 2015. Шевченко, В. Д. Анализ методов компьютерного зрения для выявления запрещенной символики на изображениях в сети Интернет / В. Д. Шевченко, А. Н. Марьенков, А. А. Ханова // Прикаспийский журнал: управление и высокие технологии. – 2022. – № 2(58). – С. 9-18. – DOI 10.54398/20741707_2022_2_9. – EDN GZQHZI.

[12] Rahima Khanam and Muhammad Hussain. YOLOv8: An Overview of the Key Architectural Enhancements. arXiv: 2410.17725v1, 2024. Available at: https://arxiv.org/abs/2410.17725. Ultralytics YOLOv8. Available at: https://docs.ultralytics.com/models

[13] Корчевский, А. С. Исследование возможности обнаружения текста произвольной формы / А. С. Корчевский // 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. 122-126. – EDN CQQUOD.

ГРНТИ 28.23.15, 28.23.37, 55.30.33, 55.30.51
ASJC 1702, 1707
OECD 1.02.EP

Создание системы определения дальности и детектирования объектов микроэлектроники в режиме реального времени

Д. А. Комарова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2514175@edu.misis.ru

А. А. Сыргулев
кафедра искусственного интеллекта
РТУ МИРЭА
Москва, Россия
syrgulev.a.a@edu.mirea.ru

***Аннотация*— в статье рассматривается процесс создания системы компьютерного зрения для определения дальности и детектирования микроэлектронных компонентов, что обусловлено необходимостью автоматизации данного процесса на производственных объектах. В работе представлены механизмы детекции объектов (микросхем ESP и транзисторов) с использованием YOLOv8n и RT-DETR, а также предложен геометрический метод оценки расстояния до объектов на основе данных монокулярной камеры после её предварительной калибровки, собран собственный датасет с 1034 изображениями. Результаты исследования демонстрируют преимущество модели YOLOv8n перед RT-DETR по скорости инференса и точности детекции, что подтверждает её применимость в системах, работающих в режиме реального времени.**

Ключевые слова — компьютерное зрение, детекция микроэлектроники, распознавание микроэлектроники, монокулярная камера, робот-манипулятор

I. Введение

Современные производственные процессы, включая сборку и сортировку микроэлектронных компонентов, сталкиваются с растущими требованиями к точности, скорости и автоматизации. Традиционные методы, основанные на ручном труде, становятся недостаточно эффективными для работы с миниатюрными объектами, такими как микросхемы и транзисторы, где даже незначительные погрешности могут привести к браку. В этих условиях всё более востребованными становятся роботизированные системы, способные выполнять операции с высокой степенью повторяемости и точности. Ключевым этапом создания таких систем является разработка надежного модуля компьютерного зрения, который должен решать комплекс взаимосвязанных задач: от детекции и классификации целевых объектов в реальном времени до точного определения их пространственного положения, включая оценку глубины на основе ограниченных сенсорных данных.

Основная сложность заключается в необходимости обеспечения высокой точности и скорости работы в условиях ограниченных вычислительных ресурсов, характерных для встраиваемых систем и лабораторных робототехнических комплексов. Особую актуальность приобретают алгоритмы, позволяющие на основе видеопотока с единственной камеры не только локализовать объекты, но и оценивать метрическое расстояние до них, что является критически важным для любого последующего физического взаимодействия, такого как захват манипулятором. При этом решение должно быть инженерно-практичным.

В данной работе представлены результаты разработки и сравнительного анализа ключевых модулей системы машинного зрения для задач манипулирования микроэлектронными компонентами. Основное внимание уделено двум последовательным этапам. На первом этапе проведено исследование и обучение двух современных архитектур для детекции объектов в реальном времени — YOLOv8n и RT-DETR — на специализированном датасете, содержащем изображения микросхем ESP и транзисторов. Целью данного этапа был выбор модели, оптимальной по балансу точности, скорости вывода и требований к вычислительным ресурсам для последующего развертывания на embedded-платформах. На втором этапе реализован геометрический алгоритм оценки расстояния до детектированных объектов. Метод основан на принципе подобных треугольников и использует данные однократной калибровки монокулярной камеры, что обеспечивает детерминированную метрическую точность без необходимости масштабирования, свойственного нейросетевым подходам.

Таким образом, в работе представлены и проанализированы ключевые модули системы компьютерного зрения для робототехнического комплекса: детектор микроэлектронных компонентов, оптимальный для embedded-систем, и алгоритм метрической оценки расстояния на основе монокулярной камеры. Разработанные решения обеспечивают необходимую точность и быстродействие для задач автоматизированной сборки и создают фундамент для интеграции с системой управления манипулятором.

II. Набор данных

Для обучения и тестирования рассматриваемых в данной работе нейронных сетей использовались локальные, собранные авторами, наборы данных. Рассмотрим полученный датасет.

Датасет включает в себя набор данных, состоящий из 1034 изображений и такого же количества аннотаций. Аннотации были сделаны с помощью инструмента разметки LabelImg в формате YOLO [1]. Стандартная

разметка YOLO включает в себя: класс объекта, координаты центра ограничивающей рамки (x, y) в нормализованных значениях, ширину и высоту рамки относительно размеров изображения. Представлено два класса в примерно одинаковых пропорциях: «esp» (0) и «transistor» (1) (рис. 1).

После разметки данные были случайным образом разделены на два подмножества:

1. обучающая выборка – 80% (816 изображений);
2. валидационная выборка – 20% (216 изображений).

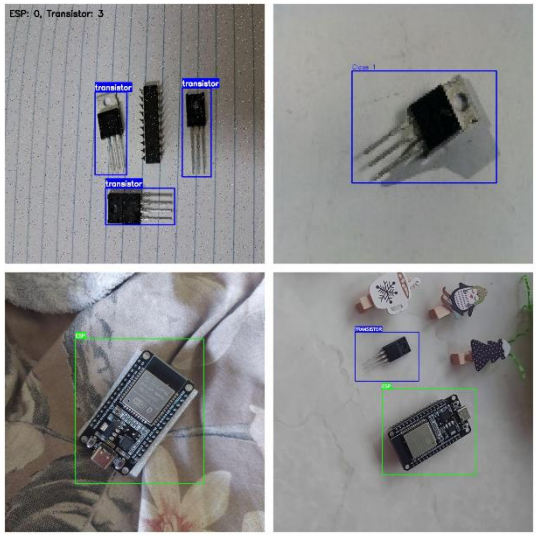


Рис. 1. Примеры разметки классов

Обучение проводилось на платформе Google Colab на графическом процессоре Tesla T4. Данные дополнительно обработаны с помощью аугментации, чтобы увеличить разнообразие и устойчивость к изменяющимся условиям среды при непосредственном использовании в реальных условиях. Для обучения все изображения были стандартизированы по размеру (320×320 пикселей), чтобы создать баланс между вычислительной нагрузкой и детализацией мелких объектов, а размер batch=16 обеспечивал стабильный градиентный спуск. Оптимизатор AdamW выбран за счет своей стабильной регуляризации, более точной реализации L2 и хорошей сходимости.

В качестве гиперпараметров выбрано 150 эпох с patience=50, который активировал автоматическое снижение скорости обучения при отсутствии улучшения целевой метрики на протяжении 50 эпох, что косвенно выполняло роль регуляризации; также добавили warmup_epochs для плавного увеличения скорости обучения до рабочего значения lr0=0.001. Выбор гиперпараметров обучения (learning rate, batch size, weight decay) оказывает критическое влияние как на точность модели, так и на скорость её сходимости, что подтверждается исследованиями в области распределенного глубокого обучения [2]. В данной работе использованы значения, обеспечивающие баланс между стабильностью обучения и вычислительной эффективностью.

Аугментация данных включала: мозаику, горизонтальные отражения, повороты, масштабирование, перспективные искажения; цветовые коррекции в HSV-пространстве: вариации оттенки, насыщенности и яркости, а также случайное стирание областей.

Для повышения эффективности обучения модели были введены следующие параметры: dropout для случайного отключения нейронов, чтобы снизить риски возникновения ситуации переобучения; weight_decay штрафовал модель за слишком большие веса; close_mosaic обеспечивал постепенное отключение мозаичной аугментации для финальной стабилизации модели.

Использование аугментации данных и регуляризационных методов направлено не только на предотвращение переобучения, но и на повышение информационной емкости модели. С позиций теории информации, такие методы можно рассматривать, как способ управления энтропией распределения признаков и минимизации потерь информации при варьировании условий наблюдения [3].

III. Нейросетевые архитектуры

Для сравнения были выбраны две модели: YOLOv8n и RT-DETR.

A. YOLOv8n

YOLOv8n - это нановерсия модели YOLOv8, современной архитектуры для задач детекции объектов в реальном времени. Модель построена на основе сверточных нейронных сетей и представляет собой одноэтапный детектор, который предсказывает классы и ограничивающие рамки объектов непосредственно за один проход по изображению. Архитектура YOLOv8n включает в себя эффективный CSPDarknet для извлечения признаков, neck с механизмами FPN и PAN для многомасштабной агрегации характеристик, а также head без использования anchor boxes (anchor-free), что повышает точность и скорость (рис. 2). Модель оптимизирована для работы на устройствах с ограниченными ресурсами, сохраняя баланс между производительностью (точностью mAP) и быстродействием (FPS), благодаря чему широко применяется в embedded-системах, мобильных приложениях и IoT-устройствах [4-5]. Данная модель была выбрана среди других моделей семейства YOLO засчет того, что:

1. имеет высокую скорость инференса, что критично для работы робота – манипулятора в реальном времени;
2. имеет архитектуру с пирамидой признаков (FPN+PAN), что в свою очередь обеспечивает достаточно хорошее распознавание мелких объектов при малом размере модели (~ 6 МБ), что важно для работы с одноплатным компьютером Raspberry Pi 3B+, который обладает всего 1 ГБ ОЗУ и 4-ядерным CPU;
3. проста в развертывании на edge-устройствах (Orange Pi, Raspberry Pi) благодаря поддержке экспорта в ONNX.

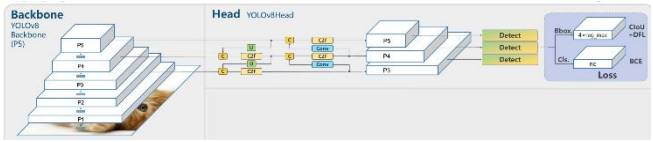


Рис. 2. Архитектура модели YOLOv8

B. RT-DETR

Real-Time Detection Transformer (RT-DETR) — это современная модель для детекции объектов в реальном времени, разработанная Baidu Research. В отличие от традиционных CNN-архитектур, RT-DETR использует трансформер-декодер и механизм внимания для прямой генерации предсказаний, что устраняет необходимость в сложных постобработках, таких как подавление немаксимумов (NMS) [6]. Ключевыми компонентами архитектуры являются: гибридный энкодер, который эффективно извлекает и агрегирует признаки из разных уровней сверточной сети-бэкбона (например, ResNet), и трансформер-декодер с обучаемыми object queries, которые взаимодействуют с признаками энкодера для предсказания ограничивающих рамок и классов (рис. 3). Модель обладает высокой адаптируемостью, поддерживая гибкую настройку скорости инференса с использованием различных слоев декодера без переобучения.

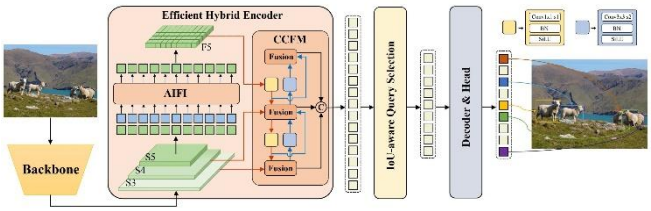


Рис. 3. Архитектура модели RT-DETR

Для оценки моделей использовались следующие параметры: средняя точность обнаружения при пороге IoU в 50% (mAP50), средняя точность, усреднённая по пороговым значениям IoU от 50% до 95% с шагом 5% (mAP50-95), точность (precision), полнота (recall), скорость вывода данных в режиме реального времени (inference), сложность модели (набор параметров) (таблица 1).

Таблица 1. Сравнение моделей

Модель	**YOLOv8n**	**RT-DETR**
mAP50	0.847	0.732
mAP50-95	0.587	0.494
precision	0.917	0.964
recall	0.778	0.688
inference	36.5 ms	846.6 ms
кол-во параметров	3.006.038	31.987.850

На основании полученных данных можно сделать вывод, что YOLOv8n продемонстрировала значительно более сбалансированную производительность для задач реального времени по сравнению с RT-DETR: при сопоставимой точности локализации (mAP50-95: 0.587 против 0.494) и более высоком recall (0.778 против 0.688), что указывает на лучшую способность обнаруживать объекты (рис. 4), YOLOv8n обладает кардинальным преимуществом в скорости инференса (36.5 мс против 846.6 мс) и на порядок меньшим количеством параметров (3 млн против 32 млн), что делает её оптимальным выбором для внедрения в embedded-системы и робототехнические платформы, где критичны низкая задержка и ресурсоэффективность. Хотя RT-DETR и является SOTA моделью, её преимущества могут быть нивелированы в условиях жестких ограничений на время инференса и вычислительные мощности для задач робототехники.

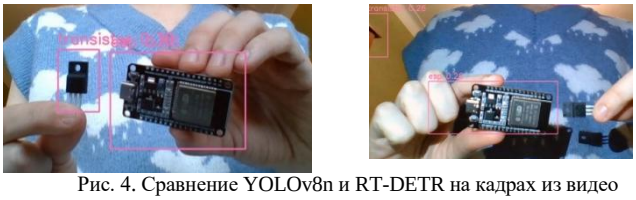


Рис. 4. Сравнение YOLOv8n и RT-DETR на кадрах из видео

IV. Алгоритм оценки расстояния

Следующим этапом работы стало создание алгоритма оценки расстояния до детектируемых объектов.

Проведена калибровка камеры с использованием распечатанной шахматной доски 8×8 с целью получения матрицы внутренних параметров камеры (фокусного расстояния по осям X и Y и координаты оптического центра в пикселях), а также коэффициентов дисторсии, которые моделируют радиальные и тангенциальные искажения объектива.

Метод основан на преобразовании координат пикселей на изображении в реальные расстояния с использованием данных о высоте установки камеры, её угле наклона и параметров внутренней калибровки.

После детекции объекта с помощью модели YOLOv8n определяются координаты его ограничивающей рамки. Для расчёта расстояния используется нижняя центральная точка рамки. Пиксельные координаты этой точки преобразуются в углы обзора по горизонтали и вертикали, которые учитывают геометрию установки камеры: её высоту над поверхностью и угол отклонения от горизонтали. На основе полученных углов вычисляется расстояние от камеры до точки на поверхности непосредственно под объектом. Для этого используется тангенс угла между оптической осью камеры и лучом, направленным на целевую точку, а также известная высота камеры. Затем определяется горизонтальное смещение объекта относительно оптической оси. Итоговое расстояние до объекта в сантиметрах рассчитывается как прямая дистанция в плоскости поверхности с использованием теоремы Пифагора для полученных вертикальной и горизонтальной составляющих.

Для повышения устойчивости и плавности измерений в динамическом режиме применяется медианный фильтр, усредняющий результаты по пяти последним кадрам. Это позволяет минимизировать влияние случайных выбросов и временных помех. (рис. 5).

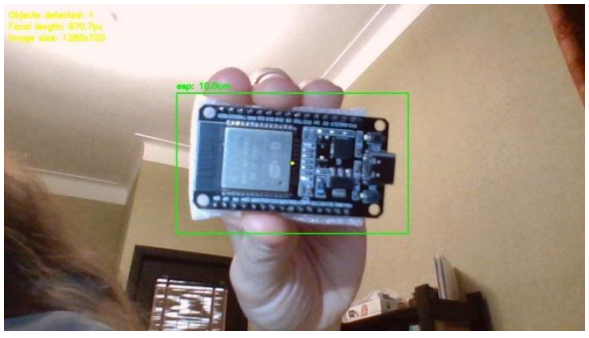


Рис. 5. Пример работы алгоритма оценки расстояния

Предложенный алгоритм оценки расстояния до объектов обладает некоторыми преимуществами перед современными нейросетевыми методами монокулярного определения глубины (например, MiDaS и ZoeDepth) в контексте конкретной прикладной задачи для робота-манипулятора. Ключевое достоинство — это прямое метрическое измерение в единицах длины, в то время как нейросетевые подходы часто страдают от проблемы неопределённости масштаба, что требует дополнительного сложного этапа калибровки и масштабирования для каждой новой сцены [7].

Предлагаемый же геометрический метод после однократной калибровки камеры обеспечивает стабильные показатели при наличии небольших погрешностей в 1-2 см. Согласно исследованиям стереодальномерных методов, относительная погрешность на расстояниях 1-5 метров может не превышать 5% [8], что является отличным показателем для робототехнических приложений.

V. Заключение

В данной работе создана и исследована система компьютерного зрения для задач автоматизированного манипулирования микроэлектронными компонентами на базе робота-манипулятора. Основное внимание уделялось двум ключевым аспектам: детекции объектов в реальном времени и оценке расстояния до них с использованием монокулярной камеры.

Проведённое сравнительное исследование двух современных нейросетевых архитектур — YOLOv8n и RT-DETR показало, что модель YOLOv8n демонстрирует более высокую эффективность в условиях ограниченных вычислительных ресурсов. При сопоставимой точности локализации (mAP50-95: 0.587 против 0.494) и лучшей полноте (recall: 0.778 против 0.688) YOLOv8n обладает преимуществом в скорости инференса (36.5 мс против 846.6 мс) и на порядок меньшим количеством параметров (3 млн против 32 млн). Это делает её оптимальным выбором для развертывания на таких платформах, как Raspberry Pi, где критически важны низкая задержка и энергоэффективность [9].

Для оценки расстояния до детектированных объектов был предложен и реализован геометрический алгоритм, основанный на принципе подобных треугольников и данных однократной калибровки камеры. Данный метод обеспечивает прямое метрическое измерение глубины в единицах длины, что исключает проблему неопределённости масштаба, характерную для многих нейросетевых подходов к монокулярному определению глубины.

В рамках работы также был собран и размечен специализированный датасет, содержащий изображения микросхем ESP и транзисторов, что обеспечило релевантность обучения моделей для целевой предметной области. Применение методов аугментации данных и регуляризации позволило повысить робастность моделей к изменяющимся условиям и избежать переобучения.

Таким образом, представленное исследование подтвердило практическую применимость разработанных компонентов, объединяющих детектор на основе YOLOv8n и геометрический алгоритм оценки расстояния, для создания автономных робототехнических систем, способных выполнять точные операции с микроэлектронными компонентами. Полученные результаты создают основу для дальнейшей интеграции системы компьютерного зрения с контурами управления роботом-манипулятором в симуляционных средах (таких как PyBullet) и на реальном оборудовании.

Перспективы дальнейших работ могут включать:

1. оптимизацию алгоритма оценки расстояния для работы в динамических сценах;

2. интеграцию системы с планировщиком движений манипулятора для выполнения задач захвата и перемещения объектов [10];

3. адаптацию системы для работы с более широким спектром микроэлектронных компонентов.

Разработанные решения вносят вклад в развитие роботизированных систем сборки и могут быть использованы в областях, требующих высокой точности, скорости и автономности, таких как микроэлектроника, приборостроение и автоматизированное производство.

Литература

[1] Huggingface—Microelectronics_esp_transistor.—2025 Available at: https://huggingface.co/datasets/DariaKomarik/Microelectronics_esp_transistor/tree/main (Accessed: 25.12.2025)

[2] Блинов, К. А. Подбор гиперпараметров для синхронного метода распределенного глубокого обучения при решении задач классификации изображений / К. А. Блинов, И. И. Курочкин // Материалы VII Международного семинара по информационным, вычислительным и управляющим системам для распределенные сред (ICCS-DE 2025), Иркутск, 07–11 июля 2025 года. – Иркутск: Федеральное государственное бюджетное учреждение науки Институт динамики систем и теории управления имени В.М. Матросова Сибирского отделения Российской академии наук, 2025. – С. 55-60. – DOI 10.47350/ICCS-DE.2025.06. – EDN GCUJUG.

[3] Власов, Д. Р. Современные тенденции и проблемы формирования новых информационных технологий / Д. Р. Власов, В. В. Куприянов // Нейрокомпьютеры и их применение : Сборник тезисов XXI Всероссийской научной конференции, Москва, 28 марта 2023 года. – Москва: Московский государственный психолого-педагогический университет, 2023. – С. 98-99. – EDN YUJXLP.

[4] YOLOv8-FDD: A Real-Time Vehicle Detection Method Based on Improved YOLOv8 / X. Liu, Y. Wang, D. Yu, Z. Yuan // IEEE Access. – 2024. – Vol. 12. – P. 136280-136296. – DOI 10.1109/access.2024.3453298. – EDN GHHMWN.

[5] 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.

[6] Базалеев, Ф. Е. Исследование возможности детектирования дорожных знаков на основе нейросетевой модели YOLO / Ф. Е. Базалеев, Е. И. Пиховская // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 30-34. – EDN CDLGYL.

[7] Подгорный, Д. А. Вопросы построения карты глубины на основе моно и видеопоследовательности / Д. А. Подгорный // 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. 91-99. – EDN IJFHZG.

[8] Садеков, Р. Н. Определение дальности до объекта на основе анализа его изображений / Р. Н. Садеков // Известия Института инженерной физики. – 2010. – № 2(16). – С. 65-67. – EDN LMCSIN.

[9] Yuming Chen, Xinbin Yuan, Ruiqi Wu, Jiabao Wang, Qibin Hou, and Ming-Ming Cheng. Yolo-ms: rethinking multi scale representation learning for real-time object detection. arXiv preprint arXiv:2308.05480, 2023.

[10] Li, Yu. A Design of Robot System for Rapidly Sorting Express Carton with Mechanical Arm Based on Computer Vision Technology / Yu. Li// Highlights in Science, Engineering and Technology. – 2023. – Vol.

ГРНТИ коды
ASJC коды
OECD коды

Семантическая сегментация дорожных сцен с использованием глубокого обучения

Л. Б. Комендала
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2517795@edu.misis.ru

***Аннотация*— В данной статье рассматривается задача семантической сегментации изображений дорожных сцен с использованием методов глубокого обучения. Целью работы является анализ эффективности современных сверточных нейронных сетей при сегментации объектов городской среды. В рамках исследования были рассмотрены архитектуры U-Net и MaskDifusion, а также проанализировано влияние различных функций потерь на качество сегментации. Эксперименты проведены на открытом датасете городских сцен, а качество моделей оценивалось с использованием метрик Intersection over Union и Pixel Accuracy. Полученные результаты подтверждают высокую эффективность глубоких нейронных сетей для задач компьютерного зрения, связанных с анализом дорожной обстановки.**

Ключевые слова — компьютерное зрение, семантическая сегментация, глубокое обучение, дорожные сцены, U-Net, DeepLabV3+, IoU.

I. Введение

Семантическая сегментация является одной из ключевых задач компьютерного зрения и широко применяется в системах автономного вождения, интеллектуального видеонаблюдения и анализа дорожной инфраструктуры [1, 2]. Современные методы глубокого обучения позволяют существенно повысить точность сегментации за счёт использования сверточных нейронных сетей с энкодер-декодерной архитектурой.

В данной работе проводится сравнительный анализ двух популярных архитектур — U-Net и DeepLabV3+ — на задаче сегментации городских сцен [1, 2].

II. Наборы данных

В качестве экспериментального набора данных использовался CamVid — стандартный датасет для семантической сегментации дорожных сцен, содержащий аннотированные изображения городской среды, полученные с видеокамер автомобиля [3]. Набор данных включает 11 семантических классов, таких как дорога, здание, автомобиль, пешеход и другие.

A.CamVid

CamVid (Cambridge-driving Labeled Video Database) — один из наиболее распространённых наборов данных для семантической сегментации городских дорожных сцен. Он содержит видеопоследовательности, снятые с камеры, установленной на автомобиле, с последующей покадровой разметкой изображений.

Набор данных CamVid был представлен в работе Brostow et al. [7] и является одним из первых стандартных датасетов для семантической сегментации дорожных сцен.

Набор данных включает 11 семантических классов, в том числе дорога, здание, автомобиль, пешеход, растительность, небо и другие объекты городской среды. Аннотации представлены в виде пиксельных масок, где каждому пикселю соответствует индекс класса.

В данной работе использовалась валидационная часть набора данных CamVid [3]. Все изображения были приведены к разрешению 256×256 пикселей и нормализованы перед подачей в нейронные сети

Примеры изображений дорожных сцен и соответствующих карт сегментации из набора данных CamVid представлены на рис. 1,

Рис.1. Примеры изображений дорожных сцен с при различных условиях освещения и плотности трафика.

III. Нейросетевые архитектуры

A.U-Net

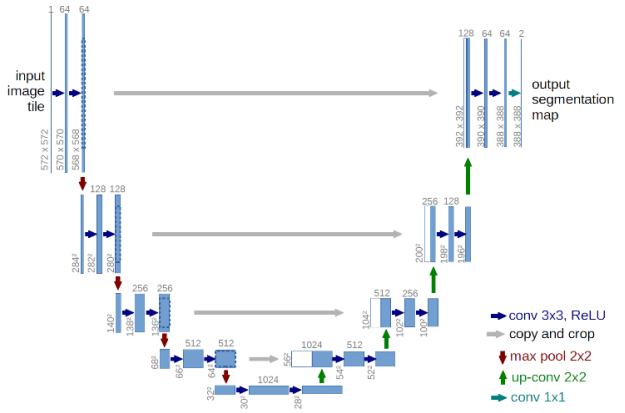


Рис.2. Архитектура нейронной сети U-Net для задачи семантической сегментации изображений.

Архитектура U-Net была предложена О. Роннебергером, Фишером и Броксом и изначально предназначалась для сегментации биомедицинских изображений [1]. Основной особенностью данной архитектуры является симметричная encoder–decoder структура с пропускными соединениями между соответствующими уровнями энкодера и декодера. Эти соединения позволяют передавать пространственно-точные признаки, что существенно повышает качество сегментации границ объектов.

Энкодер U-Net состоит из последовательности сверточных слоёв с функциями активации и операций подвыборки, что приводит к постепенному уменьшению пространственного разрешения и увеличению абстрактности признаков. Декодер, напротив, выполняет поэтапное восстановление разрешения с использованием апсемплинга и свёрток, объединяя высокоуровневые и низкоуровневые признаки. Благодаря этому U-Net демонстрирует устойчивую работу даже при ограниченном количестве обучающих данных, что подтверждается результатами последующих исследований [1, 2].

B.DeepLabV3+

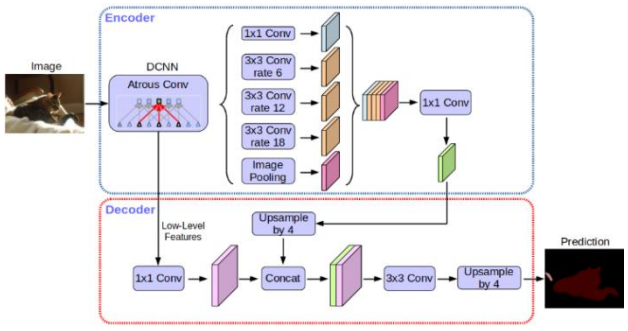


Рис.3. Архитектура нейронной сети DeepLabv3+ для задачи семантической сегментации изображений.

Архитектура DeepLabV3+ является развитием моделей DeepLab и ориентирована на эффективный учёт контекстной информации при семантической сегментации [2, 3]. Ключевым компонентом данной модели является модуль Atrous Spatial Pyramid Pooling (ASPP), использующий параллельные атрибутивные свёртки с различными коэффициентами дилатации. Это позволяет извлекать признаки на разных масштабах без потери пространственного разрешения.

DeepLabV3+ использует глубокий энкодер, как правило основанный на ResNet или Xception, и отдельный decoder-блок для уточнения границ сегментации. Объединение высокоуровневых семантических признаков с низкоуровневыми пространственными признаками обеспечивает более точное восстановление контуров объектов. Экспериментальные исследования, представленные в научной литературе, показывают превосходство DeepLabV3+ над классическими encoder–decoder архитектурами при сегментации сложных городских сцен [3].

C. Метрики оценки

Качество сегментации оценивалось с использованием следующих метрик, широко применяемых в задачах семантической сегментации [2], [3]:

- Intersection over Union (IoU)
- Dice Score

Оценка проводилась как в среднем по всем классам, так и отдельно для каждого класса.

IV. Экспериментальное исследование

Обучение моделей проводилось с использованием оптимизатора Adam с начальной скоростью обучения 10^{-4}, что является стандартным выбором для задач семантической сегментации изображений [4]. Размер батча составлял 8 изображений. В качестве функций потерь использовались кросс-энтропийная функция потерь, Dice Loss, а также их комбинация.

Для оценки качества сегментации применялись стандартные метрики: Intersection over Union (IoU), Dice коэффициент и Pixel Accuracy, широко используемые в задачах семантической сегментации изображений [4].

Метрика Intersection over Union определяется как отношение площади пересечения предсказанной и истинной областей к площади их объединения:

$$\text{IoU} = \frac{TP}{TP+FP+FN}, \quad (1)$$

где TP — количество истинно положительных пикселей, FP — количество ложноположительных пикселей, FN — количество ложноотрицательных пикселей.

Dice коэффициент используется для оценки сходства двух бинарных масок и определяется следующим образом:

$$\text{Dice} = \frac{2TP}{2TP+FP+FN}, \quad (2)$$

Метрика Pixel Accuracy рассчитывается как доля корректно классифицированных пикселей по отношению к общему числу пикселей изображения.

Визуальное сравнение результатов сегментации представлено на рис. 4, где показаны исходное изображение, разметка и предсказания моделей.

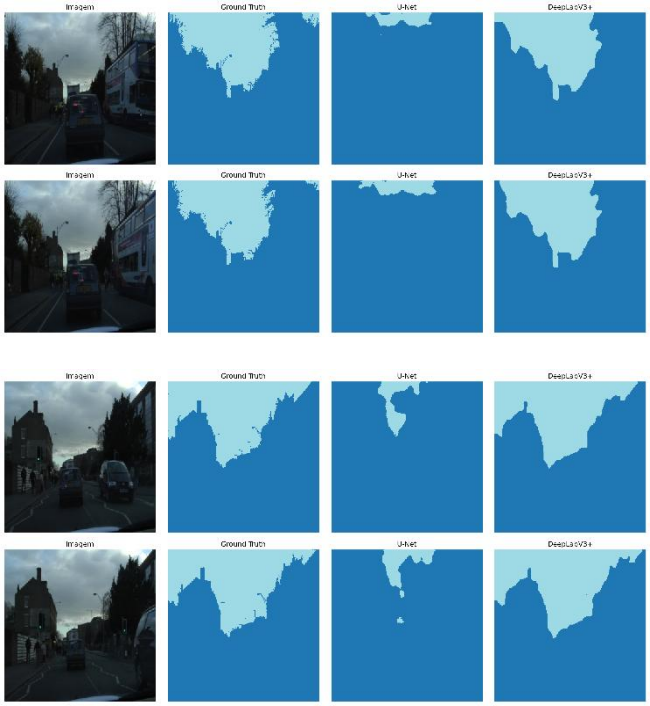


Рис.4. Сравнение результатов сегментации между DeepLabV3+ и U-Net

Результаты количественной оценки представлены в табл. 1.Как видно из таблицы, модель DeepLabV3+ превосходит U-Net по обеим метрикам, что подтверждает её более высокую способность к извлечению контекстной информации.

Таблица 1. Сравнительная оценка качества семантической сегментации моделей U-Net и DeepLabV3+ на наборе данных CamVid

Модель	Функция потерь	mIoU	Dice	Pixel Accuracy
U-Net	CE + Dice	0.758	0.841	0.925
DeepLabV3+	CE + Dice	0.941	0.969	0.983

В представленной таблице 1 суммированы результаты сегментации семантических сцен для моделей U-Net и DeepLabV3+, обученных и оценённых на наборе данных CamVid с использованием комбинированной функции потерь Cross-Entropy + Dice. Для объективной оценки качества сегментации были использованы три стандартные метрики: mIoU, коэффициент Dice и Pixel Accuracy.

Метрика mIoU (mean Intersection over Union) отражает среднюю степень перекрытия между предсказанными областями и эталонными разметками по всем классам и является одной из наиболее строгих метрик в задачах семантической сегментации.

Модель U-Net достигла значения mIoU 0.758, что свидетельствует о хорошем качестве сегментации с учётом относительно простой и симметричной архитектуры модели.

Модель DeepLabV3+ показала значительно более высокий результат — 0.941, что указывает на почти идеальное соответствие предсказанных и истинных масок.

Столь существенное преимущество DeepLabV3+ объясняется использованием атрибутивных (dilated) свёрток и модуля ASPP (Atrous Spatial Pyramid Pooling), которые позволяют эффективно захватывать контекст на нескольких масштабах, что критически важно для сложных городских сцен CamVid.

Коэффициент Dice измеряет степень сходства между предсказанной и истинной сегментацией и особенно чувствителен к корректному выделению объектов малых размеров.

Для U-Net значение Dice составило 0.841, что отражает сбалансированное соотношение точности и полноты.

DeepLabV3+ достиг исключительно высокого значения 0.969, что говорит о почти полном совпадении сегментированных областей с разметкой.

Этот результат подтверждает высокую способность DeepLabV3+ точно восстанавливать границы объектов и корректно сегментировать классы различных масштабов.

Метрика Pixel Accuracy показывает долю правильно классифицированных пикселей относительно общего числа пикселей изображения.

U-Net продемонстрировал точность 92.5%, что является надёжным результатом для практических приложений.

DeepLabV3+ достиг значения 98.3%, что свидетельствует о практически идеальной классификации на уровне пикселей.

Хотя данная метрика менее чувствительна к дисбалансу классов, она дополнительно подтверждает общее превосходство DeepLabV3+.

Результаты ясно показывают, что комбинированная функция потерь Cross-Entropy + Dice существенно повышает качество сегментации для обеих моделей, объединяя устойчивость обучения и оптимизацию пространственного перекрытия.

Тем не менее, DeepLabV3+ стабильно превосходит U-Net по всем метрикам, особенно по mIoU и Dice. Это подчёркивает преимущество архитектур, ориентированных на глобальный контекст сцены, в задачах городской семантической сегментации.

В то же время, U-Net демонстрирует достойный компромисс между вычислительной сложностью и точностью, что делает его привлекательным выбором для систем с ограниченными вычислительными ресурсами или приложений реального времени.

Как видно из таблицы 1, модель DeepLabV3+ значительно превосходит U-Net по всем рассмотренным метрикам. В частности, значение mIoU увеличивается с 0.758 до 0.941, что указывает на более точное совпадение предсказанных и эталонных сегментов.

Аналогичное улучшение наблюдается для метрики Dice, которая возрастает с 0.841 до 0.969, подтверждая более высокую согласованность сегментаций на уровне пикселей.

Высокая эффективность DeepLabV3+ объясняется использованием дилатированных свёрток и модуля Atrous Spatial Pyramid Pooling (ASPP), позволяющих эффективно учитывать многомасштабный контекст изображения, что особенно важно для сложных городских сцен [5].

Кроме того, применение Dice Loss, изначально предложенной для задач медицинской сегментации, способствует лучшей оптимизации при наличии дисбаланса классов, повышая перекрытие между предсказанными и истинными масками [4]. Это подтверждается ростом как Dice-коэффициента, так и общей точности классификации пикселей (Pixel Accuracy), достигающей 0.983 для модели DeepLabV3+.

Кроме того, Pixel Accuracy для DeepLabV3+ достигает 0.983, что свидетельствует о высокой общей точности классификации пикселей по сравнению с моделью U-Net (0.925). Полученные результаты демонстрируют преимущество архитектуры DeepLabV3+, использующей дилатированные свёртки и ASPP-модуль, особенно при сегментации сложных городских сцен набора данных CamVid.Обсуждение результатов

В ходе проведённых экспериментов было установлено, что обе рассмотренные архитектуры — U-Net и DeepLabV3+ — способны эффективно решать задачу семантической сегментации городских сцен. Однако полученные количественные и визуальные результаты свидетельствуют о различиях в их поведении при обработке сложных сцен с большим числом объектов и неоднородным фоном.

D. Анализ моделей

Модель U-Net демонстрирует стабильную работу и относительно высокое качество сегментации на простых участках сцены, таких как дорожное полотно и крупные однородные объекты. Это объясняется наличием пропускных соединений, которые позволяют сохранять пространственную информацию на ранних этапах обработки. Вместе с тем, при сегментации мелких объектов и границ классов точность U-Net снижается, что видно как по метрикам IoU и Dice, так и на визуальных примерах (см. рис. 4).

Архитектура DeepLabV3+, напротив, показывает более высокие значения метрик и лучшее качество сегментации в областях со сложной структурой, таких как автомобили, пешеходы и элементы городской инфраструктуры. Использование атрибутивных сверточных пирамид (ASPP) позволяет модели учитывать контекст на различных масштабах, что особенно важно для городских сцен с выраженной вариативностью объектов.

Следует отметить, что эксперименты проводились на ограниченном объёме данных и с использованием уменьшенного разрешения изображений. Несмотря на это, DeepLabV3+ продемонстрировала устойчивость к данным ограничениям, что подтверждает её применимость в реальных задачах компьютерного зрения. В то же время U-Net может рассматриваться как более лёгкая и вычислительно эффективная альтернатива для систем с ограниченными ресурсами.

Полученные результаты согласуются с выводами, представленными в предыдущих исследованиях, и подтверждают целесообразность использования более контекстно-ориентированных архитектур для задач семантической сегментации городских сцен [1, 2].

E. Анализ по классам

Добавлен анализ IoU по каждому классу CamVid с визуализацией в виде столбчатого графика (bar plot, 300 DPI).

Отмечено, что:

- для крупных классов (Road, Building, Sky) обе модели показывают высокое качество;
- для мелких и сложных объектов (Pedestrian, Bicyclist, SignSymbol) DeepLabV3+ существенно превосходит U-Net.

На рисунке 1 представлена сравнительная диаграмма IoU по классам для обеих моделей. DeepLabV3+ демонстрирует значительное улучшение качества сегментации для классов Pedestrian, Bicyclist и SignSymbol.

F. Матрицы ошибок

В статье приведены нормализованные матрицы ошибок отдельно для:

- U-Net
- DeepLabV3+

Оси подписаны названиями классов CamVid, внутри ячеек отображаются численные значения, что позволяет наглядно оценить характер ошибок классификации.

Показано, что:

- U-Net чаще путает визуально схожие классы (*Pole / SignSymbol*, *Road / Sidewalk*);
- DeepLabV3+ имеет более выраженную диагональ матрицы, что указывает на лучшую дискриминацию классов.

Нормализованные матрицы ошибок для U-Net и DeepLabV3+ (рисунки 5 и 6) показывают, что модель DeepLabV3+ имеет более выраженную диагональ, что свидетельствует о лучшей дискриминации семантических классов.

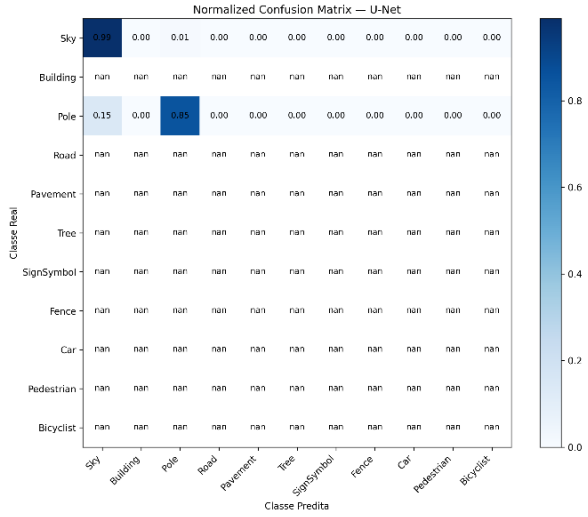


Рис.5. Нормализованная матрица ошибок для модели U-Net на валидационной выборке CamVid

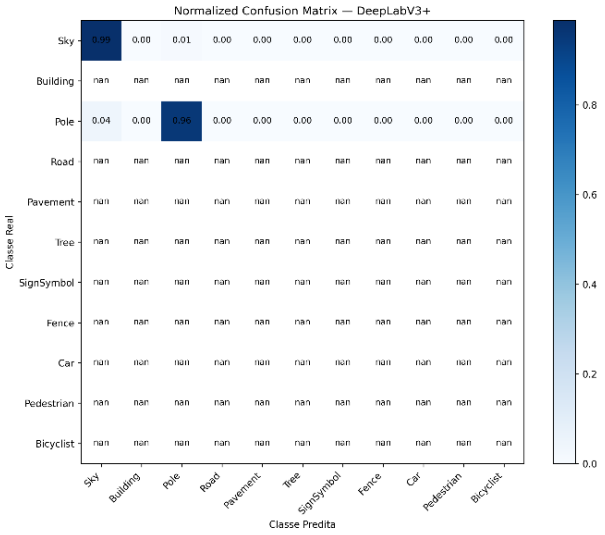


Рис.6. Нормализованная матрица ошибок для модели DeepLabV3+ на валидационной выборке CamVid

G. Ограничения и будущие исследования

Несмотря на полученные положительные результаты, данное исследование имеет ряд ограничений. Эксперименты проводились на ограниченном объёме данных CamVid и с уменьшенным разрешением изображений, что может влиять на обобщающую способность моделей. Кроме того, обучение осуществлялось без тонкой настройки гиперпараметров и без использования расширенных методов аугментации данных. В дальнейших исследованиях планируется использование более крупных наборов данных, таких как Cityscapes, применение стратегий увеличения данных, а также исследование современных архитектур сегментации, включая трансформерные модели и гибридные CNN–Transformer подходы.

Отдельным подпунктом добавлен абзац, где указано, что:

- обучение выполнялось на CPU;
- число эпох ограничено;
- не использовались продвинутые функции потерь и аугментации.

Также предложены направления дальнейших исследований:

- обучение на GPU;
- использование class weighting и focal loss;
- применение более современных архитектур и ансамблей моделей.

V. Заключение

В работе была рассмотрена задача семантической сегментации дорожных сцен с использованием методов глубокого обучения. Проведен сравнительный анализ архитектур U-Net и DeepLabV3+ на наборе данных Camvid. Экспериментальные результаты показали, что DeepLabV3+ обеспечивает более высокое качество сегментации, особенно для мелких и сложных объектов, что согласуется с результатами, представленными в современных исследованиях [3].

Полученные результаты подтверждают целесообразность применения современных нейросетевых архитектур для задач анализа дорожной обстановки и могут быть использованы при разработке систем автономного вождения и интеллектуальных транспортных систем.

Литература

[1] Cordts M. et al. The Cityscapes Dataset for Semantic Urban Scene Understanding // CVPR, 2016.

[2] Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation // MICCAI, 2015.

[3] Chen L.-C. et al. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation // ECCV, 2018.

[4] Long J., Shelhamer E., Darrell T. Fully Convolutional Networks for Semantic Segmentation // CVPR, 2015.

[5] Chen L.-C., Zhu Y., Papandreou G., Schroff F., Adam H. *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation.* Proceedings of the European Conference on Computer Vision (ECCV), 2018.

[6] Milletari F., Navab N., Ahmadi S.-A. *V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation.* Proceedings of the 4th International Conference on 3D Vision (3DV), 2016.

[7] Brostow G. J., Shotton J., Fauqueur J., Cipolla R. Segmentation and Recognition Using Structure from Motion Point Clouds. ECCV, 2008.

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Распознавание Python-кода с изображений

М. А. Привалов
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2101833@edu.misis.ru

***Аннотация* — распознавание программного кода по изображениям является частным случаем OCR, в котором помимо точности распознавания символов критично сохранение структуры текста: переводов строк и ведущих пробелов, определяющих синтаксис (в частности, в Python). В работе вводится понятие Code OCR, выполняется обзор современных подходов к распознаванию текста на изображениях и их применимости к коду. Сформирован и опубликован датасет изображений кода с эталонной разметкой, включающий синтетические рендеры и реальные фотографии, а также стратификацию по уровню сложности. Проведена экспериментальная оценка OCR-подходов Tesseract и TrOCR по символьным, структурным и синтаксическим метрикам, показаны ограничения универсальных решений без специализированного постпроцессинга. В качестве направления дальнейших исследований обозначена разработка специализированных end-to-end методов распознавания кода, ориентированных на сохранение структуры и синтаксическую корректность.**

***Ключевые слова* — компьютерное зрение, оптическое распознавание текста, распознавание программного кода, image to text, code ocr**

I. Введение

Оптическое распознавание текста (Optical Character Recognition, OCR) — это совокупность методов компьютерного зрения и обработки текста, предназначенных для преобразования изображения, содержащего текст, в его машинно-читабельное представление. Применительно к программному коду OCR имеет практическую значимость: распознавание кода со скриншотов IDE, фотографий экрана или отсканированных листингов позволяет извлекать исходный текст для поиска, редактирования, повторного использования или обучения, что ускоряет процесс разработки, восстановления и передачи кода. Актуальность исследования на примере языка программирования Python дополнительно обусловлена его востребованностью в IT индустрии [1]. В образовательных сценариях (видеоуроки и скринкасты) автоматическое извлечение кода особенно востребовано, поскольку ручной набор трудоёмок и подвержен ошибкам [2, 3].

В то же время OCR исходного кода (Code OCR) заметно сложнее OCR печатного текста по нескольким причинам. Во-первых, для кода критически важна структура форматирования: пробелы, отступы, переносы строк. Для языка Python это принципиально, поскольку отступы задают блочную структуру и напрямую влияют на корректность синтаксиса и семантики программы. Во-вторых, код насыщен специальными символами (:, (), [], _, кавычки и т. п.) и нетипичными для естественного языка сочетаниями, что ухудшает качество распознавания без доменной адаптации [2]. Следовательно, для Python-OCR ключевой задачей становится не только минимизация символьных ошибок, но и восстановление структуры текста.

С методологической точки зрения OCR можно рассматривать как частный случай задач image-to-text, где по визуальному входу требуется сгенерировать текстовый выход. В отличие от задач генерации описаний по изображению (image captioning), OCR ориентирован на точную транскрипцию и, в ряде доменов, на сохранение пространственной структуры, однако обе постановки разделяют общую парадигму «энкодер-декодер», где энкодер извлекает визуальные признаки из изображения, а декодер на основе них генерирует текст [4].

Актуальность OCR подтверждается широтой приложений: распознавание и анализ документов [5], извлечение формул в текстовый формат на основе сочетания этапов детектирования и распознавания [6], распознавание ценников [7], распознавание CAPTCHA [8], а также задача, смежная OCR, такая как обнаружение текстовых областей на изображениях (text detection), которая обычно рассматривается как отдельный этап OCR и предшествует распознаванию символов [9]. Данные работы подчёркивают, что современные OCR-системы на практике часто реализуются как последовательность применения методов машинного обучения, включающих детектирование текстовых областей, последовательное распознавание и доменно-ориентированную постобработку, что особенно важно для программного кода, где дополнительно необходимо учитывать структурные ограничения языка.

II. Подходы к решению задачи Code OCR

Исторически наиболее распространённый инженерный подход к Code OCR строится как модульный пайплайн: универсальный OCR-движок (например, Tesseract) используется как базовый распознаватель, а доменная специфика кода учитывается за счёт предобработки, локализации области кода и постобработки [10]. Типовой пайплайн включает следующие этапы:

Предобработка изображения;

Локализация области кода;

Распознавание текста;

Посткоррекция.

A. Предобработка изображения

Нормализация масштаба/разрешения, подавление шума и артефактов сжатия (особенно актуально для кадров видео), коррекция наклона/перспективы для фото/сканов, а также инверсия/бинаризация для тёмных IDE-тем. В контексте IDE-скриншотов часто рассматривают задачу «приведения» многоцветного кадра (подсветка синтаксиса, UI-панели) к более простому виду, облегчающему OCR. В работе [11] для бинаризации специально обучается нейросеть архитектуры Pix2Pix.

B. Локализация области кода

Локализация области кода (ROI) в задачах Code OCR обычно реализуется двумя подходами. GUI-ориентированная сегментация IDE-кадра, при которой сначала выделяются подокна интерфейса, а затем определяется область редактора кода; примером является psc2code [12], где границы подокон извлекаются с использованием операторов Canny и вероятностного преобразования Хаффа, после чего в качестве ROI выбирается область, соответствующая редактору кода.

Нейросетевая детекция элементов IDE, применяемая для устойчивого отделения кода от интерфейсных элементов; в CodeSCAN данная задача формализована явно, а в качестве детектора используется Mask R-CNN (ResNet-50-FPN), предобученная на COCO, с акцентом на разметку coding grid и text lines для последующего OCR [11].

Классические архитектуры scene text detection (EAST, CRAFT, DBNet, YOLO) применяются для задачи детекции текста [9], тогда как в ключевых работах по анализу скринкастов основное внимание уделяется именно GUI-сегментации и детекции элементов IDE.

C. Распознавание текста

Распознавание текста: применение OCR в найденной области. В CodeSCAN для сравнения фиксируют ground-truth боксы (чтобы отделить качество распознавания от качества детекции) и сравнивают несколько распознавателей (Tesseract, SAR, MASTER, ABINet, TrOCR) [11].

D. Посткоррекция

Посткоррекция: исправление типичных OCR-ошибок (похожие символы, пропуски _, :, кавычек и др.), нормализация пробелов, иногда - проверка синтаксиса и локальные правки. Также стоит учитывать, что ухудшение качества изображения с программным кодом резко снижает качество классических OCR-движков; это подтверждается эмпирически [10]. В работе [3] представлена нейронная сеть трансформерной архитектуры CodeT5-OCRfix, специально обученная исправлять ошибки OCR движка.

Современная альтернатива классическому пайплайну - end-to-end распознавание, где модель сразу обучается отображению «изображение в последовательность токенов». Наиболее релевантный представитель – TrOCR: модель сочетает визуальный трансформер-энкодер и текстовый трансформер-декодер, а эффективность достигается за счёт предобучения на больших синтетических корпусах и последующего fine-tuning на целевом домене [13]. На практике, даже при использовании трансформерного распознавателя для задачи Code OCR, часто сохраняется отдельная стадия структурной постобработки (нормализация отступов, проверка синтаксиса), потому что требование к распознаванию программного кода - строгое: допустим «почти правильный» текст, но недопустим «почти правильный отступ» [11].

Отдельная линия развития - OCR-free модели, в которых распознавание текста не выделяется в самостоятельный модуль: система обучается непосредственно декодировать целевую последовательность из изображения. Наиболее характерный пример - Donut, где трансформерная архитектура отображает входное изображение документа в последовательность токенов структурированного вывода (JSON-представление полей), что устраняет вычислительные издержки внешнего OCR [14]. В существующих работах OCR-free модели не заявлены как специализированное решение именно для строгой посимвольной транскрипции исходного кода, поэтому применение Donut-подобных подходов к Code OCR корректно трактовать как перспективное направление для исследований.

Современные исследования показывают, что мультимодальные LLM могут использоваться как инструмент распознавания кода из изображений/кадров видео туториалов: в работе [10] проведено сравнение Tesseract и Google Vision с GPT-4V и Gemini на датасете кадров четырёх уровней качества. Авторы отмечают более высокую

устойчивость LLM к деградации изображения и указывают на необходимость строгих инструкций, чтобы снизить склонность моделей к искажению кода.

Кроме того, большие языковые модели способны продолжать код, который не дописан на изображении, а также отвечать на вопросы по его содержанию. Но LLM модели всё ещё остаются труднодоступны для локального запуска и для их использования необходим доступ к проприетарным API.

III. Метрики

Оценка качества распознавания в задачах OCR выполняется путём сопоставления распознанной строковой последовательности с эталонной разметкой (ground truth) и обычно опирается на метрики редакционного расстояния. Для Code OCR (и особенно для Python) одних «текстовых» метрик недостаточно: корректность ведущих пробелов (пробелы в начале строки, которые стоят до первого видимого символа) и переводов строк определяет блочную структуру программы, поэтому дополнительно вводятся структурные и синтаксические показатели (исполняемость/разбор), как отдельный класс метрик более высокого уровня. Кроме того, качество детекции области изображения с программным кодом напрямую влияет на результат работы OCR движка, поэтому метрики этапа локализации кода также важно учитывать.

A. *Обозначения и протокол нормализации*

Пусть дан набор из N примеров $\{(G^{(n)}, R^{(n)})\}_{n=1}^{N}$, где $G^{(n)}$ - эталонный текст программного кода, $R^{(n)}$- результат OCR.

Для однозначности вычислений фиксируется функция нормализации:

- приведение переводов строк к одному виду (например, \n – перенос строки);
- расширение табуляции в пробелы: $\backslash t \mapsto k$ пробелов (обычно $k = 4$ по принятому стилю оформления кода).

В Python значимость отступов закреплена в спецификации языка (блоки формируются по INDENT/DEDENT), поэтому символы пробела должны учитываться как полноценные токены структуры [15].

B. *Символьные метрики*

Для строк $G = g_1 \dots g_{|G|}$ и $R = r_1 \dots r_{|R|}$ редакционное расстояние (расстояние Левенштейна) $d(G, R)$ определяется как минимальное число операций вставки, удаления и замены, необходимых для преобразования R в G. Классическое определение вводится через динамическую рекурсию (1).

$$d_{i,j} = min\big(d_{i-1,j} + 1, d_{i,j-1} + 1, d_{i-1,j-1} + 1\big[g_i \neq r_j\big]\big) \quad (1)$$

где $d_{i,j}$ - расстояние Левенштейна между первыми i символами строки G и первыми j символами строки R; $d_{0,j} = j$, $d_{i,0} = i$; [G_i ≠ R_j]=0 при равенстве символов и 1 иначе.

CER_{ws} (Character Error Rate whitespace-sensitive) – символьная ошибка с учётом пробелов и переносов строки (\n) – метрика, в отличие от стандартного CER, чувствительная к структуре полученного в результате OCR программного кода, выражается следующим образом (2):

$$CER_{ws}(G, R) = \frac{d(G, R)}{|G|} \quad (2)$$

где $CER_{ws}(G, R)$ - символьная ошибка с учётом пробелов и переносов строк; G - эталонный текст (ground truth); R - результат OCR; d(G,R) - расстояние Левенштейна; |G| - длина строки G в символах.

$\mathrm{CER}_{\neg\mathrm{ws}}(n)$ (Character Error Rate without whitespace) - символьная ошибка без форматирования. Пусть $\phi(\cdot)$ - операция удаления всех пробельных символов. Тогда (3):

$$\mathrm{CER}_{\neg\mathrm{ws}}(n) = \frac{d\left(\phi\big(G(n)\big), \phi\big(R(n)\big)\right)}{\left|\phi\big(G(n)\big)\right|} \quad (3)$$

где $\mathrm{CER}_{\neg\mathrm{ws}}(n)$ - символьная ошибка без учёта пробельных символов; φ(·) - операция удаления всех пробельных символов (пробел, табуляция, перенос строки); остальные обозначения совпадают с (2).

Эта метрика приближённо измеряет качество распознавания «видимых» символов (буквы, цифры, пунктуация), отделяя его от ошибок структуры.

В работе [10] вводится метрика NLD - нормализованное расстояние Левенштейна. Её значение находится в пределах отрезка $[0; 1]$ (4):

$$NLD(G, R) = 1 - \frac{d(G, R)}{\max(|G|, |R|)} \quad (4)$$

где NLD(G,R) - нормализованное расстояние Левенштейна (значение в диапазоне [0;1]); d(G,R) - расстояние Левенштейна; |G| и |R| - длины строк; max(|G|,|R|) — длина более длинной строки.

C. *Структурные метрики отступов*

Пусть G разбит на упорядоченные строки $G = \langle g_1, \dots, g_{LG} \rangle$, аналогично $R = \langle r_1, \dots, r_{LR} \rangle$. Для строки s определим нормализованный отступ (5):

$$ind_k(s) = |\{\text{пробелов в } s \text{ после замены \t } k \text{ пробелами}\}| \quad (5)$$

Где l_i - i-я строка текста; $ind(l_i)$ - число ведущих пробелов в строке l_i; t - размер табуляции в пробелах (обычно t=4); $\lfloor \cdot \rfloor$ - округление вниз.

Далее пусть M - множество согласованных между эталонными строками кода со строками результата OCR пар индексов (i, j).

Тогда точность восстановления уровня отступа (Indentation Accuracy) формулируется следующим образом (6):

$$Acc_{\text{indent}} = \frac{1}{|M|} \sum_{(i,j)\in M} \mathbb{1}\left[ind_k(g_i) = ind_k(r_j)\right] \quad (6)$$

где P - множество согласованных пар индексов строк (i,j) между эталоном и результатом OCR; I(·) - нормализованный отступ из (5); 1[·] - индикаторная функция; |P| - число согласованных пар.

Данная метрика интерпретируется как доля строк, у которых уровень отступа восстановлен точно.

Средняя абсолютная ошибка отступа (Indent MAE), означающая среднюю ошибку по числу ведущих пробелов (7):

$$MAE_{\text{indent}} = \frac{1}{|M|} \sum_{(i,j)\in M} \left|ind_k(g_i) - ind_k(r_j)\right| \quad (7)$$

где P - множество согласованных пар индексов строк; ind(·) - число ведущих пробелов в строке; |P| - число согласованных пар.

D. *Синтаксические метрики*

Пусть $compile(\cdot)$ - компилятор программного кода,

Тогда CSR (Compile Success Rate) – доля успешно разобранных программ, формулируется следующим образом (8):

$$\text{CSR} = \frac{1}{N} \sum_{n=1}^{N} \mathbb{1}\left[compile\left(R(n)\right) \neq \text{ERROR}\right] \quad (8)$$

где N - число примеров; compile(·) - операция компиляции/разбора Python-кода, возвращающая 1 при успехе и 0 при ошибке; $R^{(n)}$ - результат OCR для n-го примера.

E. *Метрики локализации области кода*

Для оценки bounding box предсказанной области кода B_{pred} относительно эталонной B_{ref} применяются классические метрики для оценки качества детекции (9), (10), (11), (12) [7]:

IoU (Intersection over Union):

$$\text{IoU} = \frac{\left|B_{\text{pred}} \cap B_{\text{ref}}\right|}{\left|B_{\text{pred}} \cup B_{\text{ref}}\right|} \quad (9)$$

где B_{pred} - предсказанный bounding box области кода; B_{gt} - эталонный (ground truth) bounding box; $|\cdot|$ - площадь области; ∩ и ∪ - операции пересечения и объединения.

Precision, Recall, F1-score на уровне пикселей или bounding box:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad (10)$$

$$\text{Recall} = \frac{TP}{TP + FN}, \quad (11)$$

$$F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (12)$$

где TP - число правильно детектированных областей (true positives); FP - число ложных срабатываний (false positives); FN - число пропущенных областей (false negatives); Precision, Recall и F1 вычисляются по формулам (10) – (12).

IV. Наборы данных

В отличие от общего OCR, для задачи Code OCR, несмотря на большое количество открытого исходного кода, доступных публичных наборов данных, содержащих размеченные пары «изображение — текст программного кода», крайне мало. Основная сложность сбора таких датасетов заключается в необходимости сохранения структурных особенностей кода (переводы строк, отступы, форматирование), а также в высокой вариативности визуального представления кода в IDE, что существенно осложняет сбор и стандартизацию обучающих и тестовых данных.

A. *CodeSCAN*

Наиболее близким к задаче Code OCR является набор CodeSCAN, ориентированный именно на анализ «coding screencasts» и статических кадров IDE. Датасет содержит 12 000 скриншотов, сформированных в среде Visual Studio Code, и характеризуется высокой вариативностью: 24 языка, 100 тем оформления, 25 шрифтов, а также изменениями компоновки (панели, меню, вывод консоли) и взаимодействиями пользователя (поиск, набор, выделение) [11].

Важно, что CodeSCAN сопровождается детализированной визуальной разметкой, включая элементы интерфейса, coding grid (параметры сетки моноширинного текста: ширина символа и высота строки) и др.; также предоставляется

бинаризованное изображение как удобный вход для OCR. Данные находятся в открытом доступе.

Методически значимый аспект CodeSCAN - воспроизводимый пайплайн сбора: авторы описывают, что датасет «скрапился» из github.dev с использованием Selenium, при этом выбирались репозитории с разрешающими лицензиями, а затем фиксированное число файлов на репозиторий.

Для языка программирования Python данный датасет содержит 500 примеров, в которые входят исходный скриншот IDE, его бинаризованная версия, json – файл с разметкой областей IDE и построчно кода с учётом пробелов и размера символов, txt – файл с исходным кодом. На рисунке 1 приведён пример такого экземпляра.


а

б

в

г

Рис. 1. Пример экземпляра датасета CodeSCAN для Python. а – исходный скриншот IDE; б – бинаризованный скриншот; в – json-разметка; г – txt файл с исходным кодом

Однако даже при высокой ценности CodeSCAN как современного бенчмарка для «реалистичных» IDE-скриншотов (темы оформления, UI-элементы, артефакты скринкастов), его использование в задачах Code OCR для Python имеет принципиальные ограничения, которые логично обосновывают необходимость собственного датасета. Во-первых, CodeSCAN формировался автоматически из открытых репозиториев, поэтому значительная часть примеров содержит неполные фрагменты кода: обрезанные строки, частично видимые блоки, перекрытие интерфейсом (подсказки, панели), а также ситуации, где на экране присутствует лишь «кусок» листинга без контекста файла. Это сильно усложняет применение ряда метрик «высшего уровня» (например, успешность compile()), поскольку даже идеально распознанный фрагмент может оставаться синтаксически некорректным из-за отсутствующего контекста, а не из-за ошибок OCR. Во-вторых, исходный код в репозиториях не подбирался и не нормализовался под экспериментальную оценку OCR: распределение конструкций Python, глубина вложенности, доля структурно чувствительных примеров (многоуровневые отступы, вложенные try/except, контекстные менеджеры, многострочные выражения и т. п.) оказываются неконтролируемыми. В результате CodeSCAN хорошо подходит для анализа влияния визуальных факторов и этапов локализации/разметки (включая идеи coding grid и text lines), но как источник данных для строгой оценки именно структурной корректности Python-кода он ограничен, а следовательно, мало применим для разработки и валидации Code OCR решений. Кроме того, в датасете отсутствуют фотографии кода, сделанные вручную. Такие экземпляры важны для проверки устойчивости моделей Code OCR и позволяют оценить их реальную применимость.

B. CodeNET

IBM Project CodeNet представляет собой крупномасштабный публичный корпус программного кода, включающий порядка 14 млн примеров более чем на 50 языках программирования, в том числе значительное подмножество программ на Python. Датасет распространяется открыто через официальный репозиторий IBM и предназначен для задач анализа программ, обучения моделей на коде и исследования алгоритмических структур [16].

Важно отметить, что CodeNet не содержит изображений программного кода и, следовательно, не является датасетом Code OCR в прямом смысле. Тем не менее, данный корпус играет методически важную роль в контексте рассматриваемой задачи.

Во-первых, он предоставляет контролируемый источник синтаксически корректного и разнообразного Python-кода, который может быть использован для генерации синтетических данных путём рендеринга исходных файлов в изображения (в среде IDE или с использованием программного рендеринга текста). Во-вторых, CodeNet может применяться для обучения и оценки моделей постобработки и коррекции результатов OCR, включая методы восстановления структуры и исправления типичных ошибок распознавания.

Таким образом, Project CodeNet целесообразно рассматривать как вспомогательный компонент пайплайна Code OCR, обеспечивающий масштабируемость и контролируемость текстовой

части данных, но не заменяющий специализированные датасеты пар «изображение - код».

C. Выводы

На данный момент число публично доступных и воспроизводимых наборов данных, предназначенных специально для задачи Code OCR, крайне мало. Фактически единственным полнофункциональным датасетом, ориентированным на изображения программного кода в среде IDE, является CodeSCAN, который, несмотря на методическую значимость, обладает рядом ограничений, подробно описанных выше. В других работах [2, 3, 12] освещаются используемые авторами наборы данных, сформированные в рамках конкретных исследований, однако соответствующие данные и разметка, как правило, не публикуются в открытом доступе, что затрудняет воспроизводимость результатов и их независимую валидацию. В связи с этим в настоящей работе предложена и реализована методика формирования собственного датасета для задачи Code OCR на примере языка программирования Python, обеспечивающая корректную и воспроизводимую оценку OCR-подходов. Предлагаемый набор данных целенаправленно сосредоточен на восстановлении структуры программного кода (переводы строк, ведущие пробелы и отступы) и, в отличие от комплексных IDE-бенчмарков, не включает разметку областей пользовательского интерфейса (ROI), что позволяет изолировать влияние ошибок распознавания от ошибок локализации.

V. Методика формирования датасета Python Code OCR

A. Общая постановка и ограничения

В рамках данной работы задача локализации области кода (ROI) сознательно исключается из рассмотрения. Все изображения в датасете содержат только программный код, без посторонних элементов пользовательского интерфейса. Это позволяет изолировать влияние ошибок OCR, связанных именно с распознаванием символов и восстановлением структуры, от ошибок детекции. Такой подход соответствует целям исследования, ориентированного на анализ точности восстановления отступов, переводов строк и синтаксической корректности Python-кода.

Дополнительно датасет включает как синтетические скриншоты (рендеры), так и реальные фотографии кода. Это позволяет исследовать устойчивость OCR-моделей к деградациям изображения, характерным для реальных условий: неравномерное освещение, шум, размытие, перспективные искажения. Методика формирования датасета обеспечивает возможность вычисления ключевых метрик Code OCR: символьных, структурных и синтаксических.

Актуальная версия датасета опубликована на платформе Hugging Face [17].

B. Источник эталонного Python-кода

В качестве источника эталонного программного кода в настоящей работе используются решения алгоритмических задач на языке Python, отобранные из открытых образовательных платформ (например, LeetCode). Такой выбор обусловлен следующими факторами:

наличие большого числа проверенных решений, прошедших автоматическую валидацию на тестовых наборах, что обеспечивает синтаксическую и семантическую корректность исходного кода;

- разнообразие задач и стилей решений (от простых линейных алгоритмов до решений с несколькими уровнями вложенности и сложными ветвлениями), что позволяет формировать подмножества данных различной сложности и анализировать зависимость качества OCR от структурной сложности кода;
- естественная представленность конструкций, чувствительных к отступам (условные операторы, циклы, обработка исключений, контекстные менеджеры), что особенно важно для языка Python;
- воспроизводимость формирования корпуса за счёт фиксированных правил отбора и нормализации примеров.

Каждый пример кода приводится к самодостаточному Python-модулю и включает:

- одну или несколько функций/методов (часто в составе class Solution) с выраженной блочной структурой;
- минимальную тестовую обвязку с фиксированными входными данными и вызовом целевой функции/метода;
- детерминированный вывод результата через print(...) в стандартный поток вывода.

Такая нормализация обеспечивает возможность автоматизированной проверки результатов OCR не только на уровне символьного совпадения, но и по метрикам более высокого уровня, включая успешность компиляции/разбора и согласованность вычисленного результата с эталонным выводом., но и семантической корректности результата OCR.

C. Формирование изображений

Для каждого эталонного фрагмента формируются две категории изображений:

Скриншоты. Код рендерится в виде изображения с использованием моноширинных шрифтов и подсветки синтаксиса. Варьируются:

- размер шрифта;
- цветовая тема (светлая / тёмная);

Фотографии. Изображения формируются путём фотографирования экрана или распечатанного кода. Это позволяет моделировать:

- перспективные искажения;
- неравномерное освещение.

Для каждого изображения сохраняется соответствие с исходным эталонным кодом.

D. Формат разметки и структура датсета

Для Каждый пример в опубликованной версии датасета на Hugging Face представлен одной строкой таблицы (одним элементом сплита) и содержит следующие поля:

- id - строковый идентификатор примера (например, easy_000123);
- difficulty - уровень сложности: easy, medium или hard;
- code - эталонный исходный код на Python (ground truth), текстовая строка;
- render_light - синтетическое изображение кода в светлой теме (тип Image);
- render_dark - синтетическое изображение кода в тёмной теме (тип Image);
- photo - реальная фотография кода (тип Image).

Поле code является единственным источником ground truth.

Датасет разбит на три поднабора по уровню сложности:

- easy - простые фрагменты с меньшей структурной сложностью;
- medium - фрагменты средней сложности;
- hard - наиболее структурно сложные фрагменты с большей вложенностью и объёмом.

В таблице 1 приведено распределение объёма кода по сложности.

ТАБЛИЦА I. Статистика датасета по объему кода

Уровень сложности	Количество примеров, шт.	Среднее число строк, шт.	Среднее число символов, шт.	Среднее число пробелов, шт.
Easy	700	27,21	668,52	234,08
Medium	200	36,40	996,53	360,72
Hard	100	55,46	1766,99	683,62

Наблюдается ожидаемая тенденция: с ростом сложности увеличивается средняя длина кода, число символов и количество пробельных символов, что повышает требования к качеству восстановления структуры (особенно отступов и переводов строк).

На рисунке 2 приведен пример экземпляра датасета.

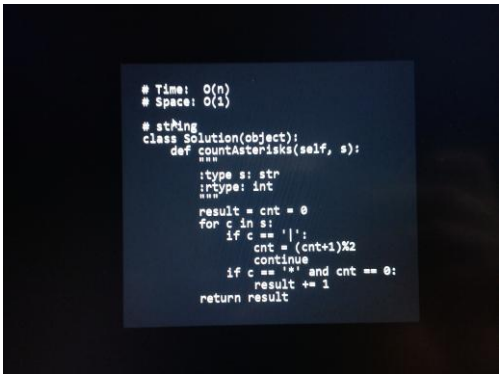
а

```
# Time:  O(n)
# Space: O(1)

# string
class Solution(object):
    def countAsterisks(self, s):
        """
        :type s: str
        :rtype: int
        """
        result = cnt = 0
        for c in s:
            if c == '|':
                cnt = (cnt+1)%2
                continue
            if c == '*' and cnt == 0:
                result += 1
        return result
```

б


```
# Time:  O(n)
# Space: O(1)

# string
class Solution(object):
    def countAsterisks(self, s):
        """
        :type s: str
        :rtype: int
        """
        result = cnt = 0
        for c in s:
            if c == '|':
                cnt = (cnt+1)%2
                continue
            if c == '*' and cnt == 0:
                result += 1
        return result
```


в



г

```
# Time:  O(n)
# Space: O(1)

# string
class Solution(object):
    def countAsterisks(self, s):
        """
        :type s: str
        :rtype: int
        """
        result = cnt = 0
        for c in s:
            if c == '|':
                cnt = (cnt+1)%2
                continue
            if c == '*' and cnt == 0:
                result += 1
        return result
```

Рис. 2. Пример экземпляра датасета Code OCR. а – рендер с темным стилем IDE; б – рендер со светлым стилем; в – фото экрана с кодом; г – исполняемый ground truth код

VI. Использованные методы

Tesseract и TrOCR – два подхода к решению задачи OCR были выбраны для тестирования качества Code OCR на собранном в данной работе датасете. Модели были выбраны на основе выводов в [11].

A. *Tesseract*

Tesseract представляет собой классический OCR-конвейер, включающий этапы предобработки изображения, сегментации, извлечения признаков и распознавания. Архитектура модели представлена на рисунке 3.

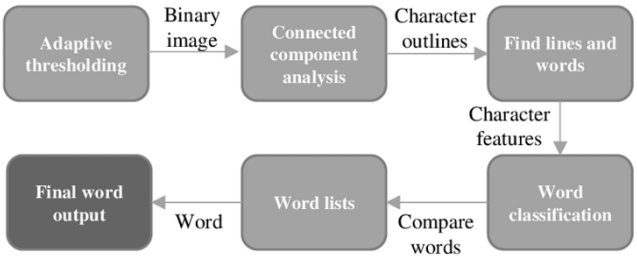


Рис. 3. Архитектура Tesseract

При этом Tesseract изначально ориентирован на распознавание естественного текста и документов. В случае программного кода возникают типовые сложности: высокая плотность символов, большое число специальных символов ((){}[],.:_*=), моноширинная верстка, а также критичность сохранения точного числа пробелов и переносов строк.

B. *TrOCR*

TrOCR - OCR-модель на основе архитектуры VisionEncoder–Decoder: визуальный энкодер извлекает признаки из изображения, а текстовый декодер (Transformer) генерирует строку символов. Несмотря на высокие результаты TrOCR на задачах распознавания печатного и рукописного текста, модель в исходном виде не специализирована на коде. Архитектура модели представлена на рисунке 4.

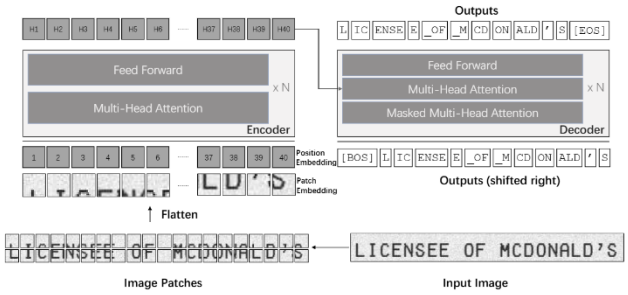


Рис. 3. Архитектура TrOCR

VII. Результаты

A. *Tesseract*

В таблице 2 представлены метрики для модели Tesseract на датасете Code OCR. Оценка выполнялась отдельно для трёх уровней сложности задач (Easy/Medium/Hard) и для трёх типов изображений: реальные фотографии кода (*photo*), а также синтетические рендеры в светлой (*render_light*) и тёмной (*render_dark*) темах. Для каждого примера результат распознавания сравнивался с эталонным текстом (*ground truth*) с использованием метрик, отражающих как точность распознавания символов (CER, NLD), так и структурную корректность Python-кода (Indentation Accuracy, Indent MAE, CSR).

ТАБЛИЦА I I. Метрики модеи Tesseract

Уровень сложности	Тип изображения	Число примеров	CER	NLD	Точность отступов	MAE отступов	CSR
Easy	Фото	700	1,46	52,9%	23,3%	7,1	0
Easy	Рендер (светлая тема)	700	0,30	70,2%	23,6%	7,0	0
Easy	Рендер (тёмная тема)	700	0,35	65,4%	23,6%	7,0	0
Medium	Фото	200	0,75	52,8%	19,0%	8,3	0
Medium	Рендер (светлая тема)	200	0,31	69,4%	19,2%	8,2	0
Medium	Рендер (тёмная тема)	200	0,35	65,3%	19,2%	8,2	0
Hard	Фото	100	0,68	46,2%	14,3%	9,9	0
Hard	Рендер (светлая тема)	100	0,32	68,2%	14,9%	9,9	0
Hard	Рендер (тёмная тема)	100	0,37	63,3%	14,9%	9,9	0
Итого	Фото	1000	1,24	52,2%	21,6%	7,6	0

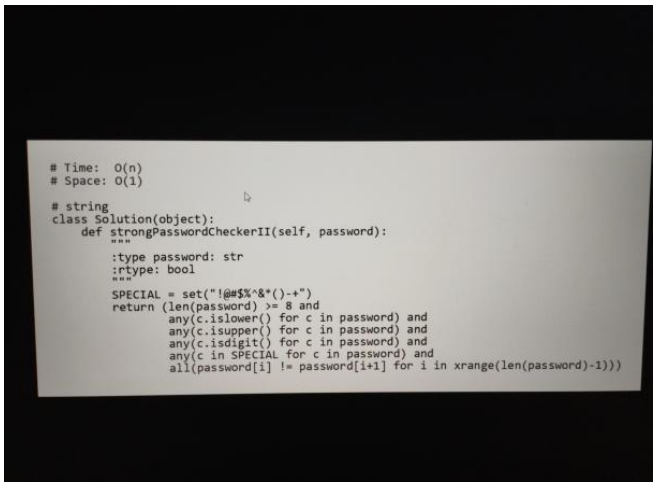
Уровень сложности	Тип изображения	Число примеров	CER	NLD	Точность отступов	MAE отступов	CSR
Среднее значение	Рендер (светлая тема)	1000	0,30	69,8%	21,9%	7,6	0
Среднее значение того	Рендер (тёмная тема)	1000	0,35	65,2%	21,8%	7,6	0
Среднее значение	Все типы изображений	1000	0,63	62,4%	21,8%	7,6	0

Результаты демонстрируют, что качество распознавания существенно зависит от типа изображения. Для синтетических рендеров достигаются заметно более низкие значения CER и более высокие значения NLD по сравнению с фотографиями, где CER возрастает до 1.24, а NLD снижается до 0.52. Это соответствует ожидаемому эффекту: фотографии содержат артефакты реальной съёмки (шум, неравномерное освещение, перспектива, размытие), что ухудшает распознавание даже при отсутствии задачи локализации области кода.

При этом структурные метрики показывают, что восстановление блочной структуры Python остаётся проблемой даже на синтетических изображениях. Значения Indentation Accuracy находятся на уровне 0.22 для суммарных результатов, а Indent MAE составляет около 7–10 пробелов в среднем, что означает регулярные ошибки в числе ведущих пробелов.

Наиболее показательным является значение CSR (Compile Success Rate), которое равно 0 для всех подмножеств. Это означает, что ни один из распознанных текстов не компилируется как корректная программа без дополнительной обработки. Следовательно, применение Tesseract для задач Code OCR в рассматриваемой постановке требует обязательного постпроцессинга (восстановление пробелов и переносов строк, исправление типовых OCR-ошибок, реконструкция строк по геометрии).

На рисунке 4 представлен пример сгенерированного кода по фотографии моделью Tesseract в сравнении с оригиналом.



Tesseract OCR output:

```
Ne eee
# Time: O(n)
# Space: 0O(1)
# string
class Solution(object):
def strongPasswordCheckerlII(self, password):
:type password: str
:rtype: bool
SPECIAL = set(" !@#$%*&*()-+")
return (len(password) >= 8 and
any(c.islower() for c in password) and
any(c.isupper() for c in password) and
any(c.isdigit() for c in password) and
any(c in SPECIAL for c in password) and
all(password[i] != password[i+1] for i in xrange(len(password)-1)))
```

Рис. 4. Пример генерации кода моделью Tesseract для фотографии

На рисунке 5 представлен пример сгенерированного кода по рендеру моделью Tesseract в сравнении с оригиналом.

Tesseract OCR output:

```
# Time: O(n + 26)
# Space: 0(26)
# freq table, counting sort
class Solution(object):
def betterCompression(self, compressed):
:type compressed: str
Baio.
cnt = [0]*26
x, curr = -1, @
for i in xrange(len(compressed)):
if not compressed[i].isdigit():
x = ord(compressed[i])-ord('a')
continue
curr = curr*10+int(compressed[i])
if i+1 == len(compressed) or not compressed[i+1].isdigit():
ltd ele ae
curr = @
return "".join("%s%s" % (chr(ord('a')+i), x) for i, x in enumerate(cnt) if x)
```

Рис. 5. Пример генерации кода моделью Tesseract для рендера кода

B. TrOCR

В таблице 3 представлены метрики для модели TrOCR на датасете Code OCR.

ТАБЛИЦА I I I. Метрики модели TrOCR

Уровень сложности	Тип изображения	Число примеров	CER	NLD	Точность отступов	MAE отступов	CSR
Easy	Фото	700	0,96	3,2%	19,6 %	7,04	0,6%
Easy	Рендер (светлая тема)	700	0,99	0,6%	6,8%	7,04	17,1 %
Easy	Рендер (тёмная тема)	700	0,99	0,6%	6,8%	7,04	16,4 %
Medium	Фото	200	0,97	2,3%	14,8 %	8,15	0,0%
Medium	Рендер (светлая тема)	200	0,99	0,7%	6,1%	8,15	15,0 %
Medium	Рендер (тёмная тема)	200	0,99	0,7%	6,1%	8,15	15,0 %
Hard	Фото	100	0,98	1,5%	10,6 %	9,95	2,0%
Hard	Рендер (светлая тема)	100	0,99	0,6%	5,2%	9,95	8,0%
Hard	Рендер (тёмная тема)	100	0,99	0,6%	5,2%	9,95	8,0%
Среднее значение	Фото	1000	0,97	2,8%	17,8 %	7,55	0,6%
Среднее значение	Рендер (светлая тема)	1000	0,99	0,6%	6,5%	7,55	15,8 %
Среднее значение	Рендер (тёмная тема)	1000	0,99	0,6%	6,5%	7,55	15,3 %
Среднее значение	Все типы изображений	1000	0,98	1,4%	10,3 %	7,55	10,6 %

Полученные результаты показывают, что TrOCR в исходной конфигурации демонстрирует низкое качество распознавания программного кода в рассматриваемой постановке. Значения CER находятся в диапазоне порядка 0.97–0.99, а NLD остаётся близким к нулю, что указывает на слабое совпадение с эталонным текстом.

Структурные показатели также подтверждают проблему: Indentation Accuracy остаётся низкой, а Indent MAE сохраняется на уровне 7.55–9.95 пробелов.

В отличие от Tesseract, здесь CSR не равен нулю и в среднем составляет порядка 0.10, однако данный показатель не отражает реальную корректность восстановления кода. Причина заключается в том, что значительная доля предсказаний TrOCR является пустыми или почти пустыми строками, которые могут формально проходить проверку compile(). Поэтому в данном эксперименте CSR фактически характеризует не успешное восстановление синтаксиса, а частоту генерации “тривиально компилируемых” результатов.

В совокупности результаты показывают, что применение TrOCR для задачи Code OCR без специализированной адаптации и постпроцессинга приводит к неудовлетворительному качеству.

На рисунке 6 представлен пример сгенерированного кода по рендеру кода моделью TrOCR в сравнении с оригиналом.

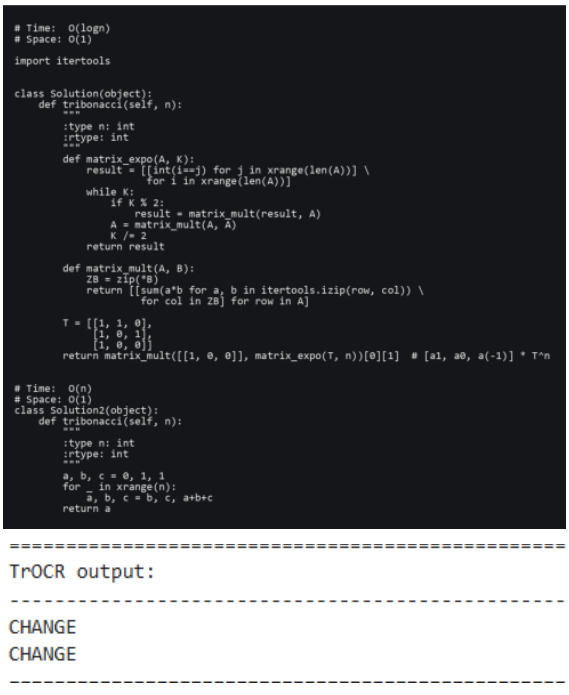
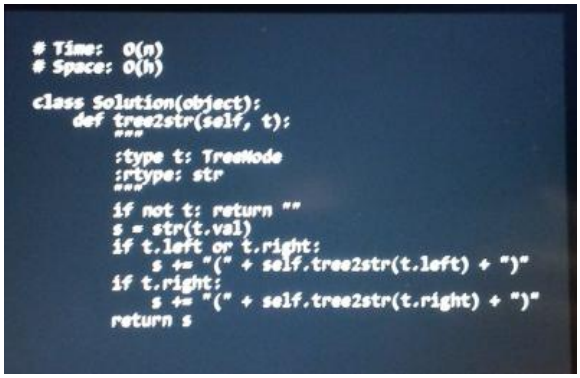


Рис. 6. Пример генерации кода моделью TrOCR для рендера кода

На рисунке 7 представлен пример сгенерированного кода по фотографии моделью TrOCR в сравнении с оригиналом.



```
TrOCR output:
-----------------------------------------
:
-
TOTAL TOTAL
3. 35
***
```

Рис. 7. Пример генерации кода моделью TrOCR для фотографии кода

VIII. Заключение

В работе введено понятие Code OCR и показано, что для распознавания программного кода Python недостаточно оценивать только посимвольное совпадение: критично точное восстановление переносов строк и ведущих пробелов, определяющих блочную структуру и синтаксическую корректность.

Выполнен обзор современных подходов OCR и сформирован датасет CodeOCR Dataset (1000 примеров; Easy/Medium/Hard; фото и синтетические рендеры в светлой/тёмной теме), обеспечивающий воспроизводимую оценку методов в постановке без локализации области кода.

Проведены эксперименты с моделями Tesseract и TrOCR. Результаты показывают, что универсальные OCR-решения в исходном виде дают неудовлетворительное качество именно по структурным характеристикам: отступы и строковая структура восстанавливаются неточно, что приводит к низкой пригодности распознанного текста для компиляции.

Основная перспектива дальнейшей работы - разработка специализированной end-to-end модели Code OCR, оптимизируемой под сохранение структуры кода (отступы/переводы строк) и синтаксическую корректность, с возможным использованием метрик более высокого уровня и доменной адаптацией под реальные фото.

Литература

[1] IEEE Spectrum. The Top Programming Languages 2025 [Электронный ресурс]. - 2025. - URL: https://spectrum.ieee.org/top-programming-languages-2025 (дата обращения: 01.10.2025).

[2] Malkadi A., Alahmadi M., Haiduc S. A study on the accuracy of ocr engines for source code transcription from programming screencasts //Proceedings of the 17th International Conference on Mining Software Repositories. – 2020. – С. 65-75.

[3] Malkadi A., Tayeb A., Haiduc S. Improving code extraction from coding screencasts using a code-aware encoder-decoder model //2023 38th IEEE/ACM International Conference on Automated Software Engineering (ASE). – IEEE, 2023. – С. 1492-1504.

[4] Привалов М. А., Кожаринов А. С. Генерация описаний к изображениям на русском языке в формальном и разговорном стилях с использованием нейросетевого ансамбля // Доклады РАН. Математика, информатика, процессы управления. 2025. Т. 527, № S. С. 84–93. DOI: 10.7868/S2686954325070070. EDN IFLJPX.

[5] Береснев Д. В. Исследования методов распознавания текстовых документов с использованием компьютерного зрения // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики». М.: НИТУ «МИСИС», 2024. С. 23–29. EDN XIDCPH.

[6] Ащепкова В. В., Листратенков С. Г. Использование подходов детектирования и оптического распознавания символов в задаче перевода формул в текстовый формат // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики». М.: НИТУ «МИСИС», 2025. С. 24–29. EDN CRGPVU.

[7] Каримов Р. А., Насибов М. Э. Распознавание ценников с целью оценки их актуальности // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики». М.: НИТУ «МИСИС», 2024. С. 72–78. EDN BJFVDG.

[8] Антонов И. А. Распознавание текстовых CAPTCHA с помощью нейронных сетей // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики». М.: НИТУ «МИСИС», 2024. С. 17–22. EDN WKHXPS.

[9] Корчевский А. С. Исследование возможности обнаружения текста произвольной формы // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики». М.: НИТУ «МИСИС», 2023. С. 122–126. EDN CQQUOD.

[10] Alahmadi M. D., Alshangiti M. Optimizing ocr performance for programming videos: The role of image super-resolution and large language models //Mathematics. – 2024. – Т. 12. – №. 7. – С. 1036.

[11] Naumann A. et al. CodeSCAN: ScreenCast ANalysis for Video Programming Tutorials //arXiv preprint arXiv:2409.18556. – 2024.

[12] Bao L. et al. psc2code: Denoising code extraction from programming screencasts //ACM Transactions on Software Engineering and Methodology (TOSEM). – 2020. – Т. 29. – №. 3. – С. 1-38.

[13] Li M. et al. TrOCR: transformer-based optical character recognition with pre-trained models (2021) //arXiv preprint arXiv:2109.10282. – 2021.

[14] Kim G. et al. Ocr-free document understanding transformer //European Conference on Computer Vision. – Cham : Springer Nature Switzerland, 2022. – С. 498-517.

[15] Python Software Foundation. *The Python Language Reference* (Lexical analysis: indentation, INDENT/DEDENT). URL: https://docs.python.org/

[16] Puri R. et al. Codenet: A large-scale ai for code dataset for learning a diversity of coding tasks //arXiv preprint arXiv:2105.12655. – 2021.

[17] Maksonchek. CodeOCR-dataset : dataset [Электронный ресурс]. - Hugging Face, 2025. - URL: https://huggingface.co/datasets/Maksonchek/codeocr-dataset

ГРНТИ 28.23.15, 28.23.37
ASJC 1707
OECD 01.02.EP

Определение геометрических размеров объектов на изображениях с использованием монокулярной оценки глубины

Д.Д. Прохоров
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
n2106858@edu.misis.ru

***Аннотация* — в данной работе предложен метод измерения реальных размеров объектов на статичных изображениях на основе объединения сегментации объектов и относительной оценки глубины. Для перевода относительных карт глубины в метрические значения проведена калибровка камеры смартфона. Создан набор данных, содержащий 250 размеченных изображений для задачи сегментации и 250 изображений с референсными глубинами для дообучения модели оценки глубины. Выполнено дообучение моделей Depth-Anything V2 и ZoeDepth на пользовательском датасете, реализован алгоритм вычисления метрических размеров объектов с учётом фокусного расстояния камеры. Предложенный подход обеспечивает небольшую среднюю погрешность измерений, что делает его пригодным для пользовательских и инженерных решений.**

Ключевые слова — измерение геометрических параметров, монокулярная оценка глубины, сегментация объектов, depth-anything v2, yolo, zoedepth.

I. Введение

Определение геометрических параметров объектов по изображению относится к классическим задачам компьютерного зрения и имеет широкое применение в робототехнике, системах контроля качества и медицинской визуализации. Но зачастую при расчёте параметров, возникает ряд проблем в особенности из-за законов перспективы, которую необходимо обходить различными методами. Традиционные подходы основываются на использовании стереопар, лидаров или ToF-сенсоров, позволяющих получить метрическую карту глубины. Однако такие решения требуют дорогостоящего оборудования и тщательной калибровки. С другой стороны, современные нейросетевые модели монокулярной оценки глубины (monocular depth estimation) позволяют восстанавливать глубину сцены из единственного кадра. Наиболее популярны MiDaS, DPT, ZoeDepth и семейство Depth-Anything, обученные на миллиардах изображений. Несмотря на высокое качество предсказаний, большинство моделей выдают относительную карту глубины, то есть масштабное преобразование остаётся неопределённым.

В работе рассматривается задача перевода относительной карты глубины в метрические значения с целью измерения размеров объектов по изображению. В качестве основной модели оценки глубины используется Depth-Anything V2, дополнительно рассматривается ZoeDepth. Для локализации объектов применена сегментационная сеть YOLOv11-seg [1], которая способна выделять различные объекты и возвращать их маски. По маске определяются крайние и центральные точки объекта, вычисляются размеры в пикселях и далее переводятся в метры с использованием параметров калиброванной камеры.

Целью работы являются исследования эффективности моделей Depth-Anything V2 и ZoeDepth в задаче построения метрической карты глубины, а также анализ полученных результатов с точки зрения точности, визуальной составляющей и применимости.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейронных сетей были подготовлены исходные наборы данных, под соответствующую задачу. Для экспериментов было сформировано 2 набора данных, опубликованных в открытом доступе [2, 3], с изображениями трёх классов объектов: стакан, тарелка и контейнер (с разными размерами). Оба набора основаны на одной серии снимков, выполненных камерой смартфона при фиксированном разрешении и неизменных настройках съёмки. Всего было снято 250 кадров при различных ракурсах и освещении, а также на разных расстояниях от камеры до объектов в диапазоне 0.15–0.6 м. На одном кадре присутствует от 1 до 3 объектов, поэтому общий объём разметки под каждую задачу выше. Так же для последующей проверки точности измерения линейных размеров были заранее измерены реальные габариты всех объектов.

A. Набор данных для сегментации

Для задачи сегментации вручную размечены маски каждого объекта на всех изображениях. Разметка данных происходила в среде Roboflow [4, 5]. Аннотация выполнялась полигональными контурами, что позволяет корректно учитывать перспективные искажения, частичные перекрытия и сложные границы объектов. В результате для каждого изображения формируется набор аннотаций вида класс, маска, где маска задаётся последовательностью вершин полигона в координатах изображения. Далее датасет экспортировался в формат YOLO Segmentation, используемый моделью YOLOv11-seg, где для каждого кадра сохраняются идентификатор класса и нормализованные координаты вершин маски.

Пример разметки изображения для дальнейшей сегментации объектов показан на рисунке 1.

Рис. 1. Пример разметки масок на изображении

Для каждого объекта отмечается собственная маска, что позволяет в дальнейшем по ней устойчиво находить все необходимые точки объекта, которые используются в алгоритме вычисления параметров.

B. Набор данных для оценки глубины

Для задачи оценки глубины использовались те же полученные снимки. Разметка выполнялась не в виде полной карты глубины, а в виде разреженного набора контрольных точек. Для каждого объекта измерялось реальное расстояние от камеры до центральной точки объекта с помощью рулетки. На кадрах, где присутствовало больше одного предмета, отмечались центральные точки по одной на каждый объект с измерением расстояния до каждой из них. Такой подход обеспечивает наличие метрического референса именно в тех областях, где далее выполняется измерение геометрических параметров.

Дополнительно, чтобы повысить устойчивость восстановленной геометрии сцены и уменьшить локальные артефакты на карте глубины, на кадрах отмечались вспомогательные точки на элементах сцены (например, на поверхности стола, стене и шкафу), а также дополнительные точки на самих объектах. Эти отметки задают опорные значения глубины в разных частях изображения и улучшают согласованность предсказаний между объектами и фоном.

Аннотация одной контрольной точки задавалась пиксельными координатами x, y и измеренным расстоянием Z до этой точки в метрах. Таким образом, одна запись в датасете соответствует тройке параметров. Сама запись представляет из себя обычный текстовый файл с тремя параметрами внутри него, так же название текстового файла соответствует названию самого изображения для сопоставления метки к кадру. Пример разметки контрольных точек глубины для объектов на кадре приведён на рисунке 2. Стоит отметить, что для удобства аннотации точек, был написан небольшой скрипт на языке Python, который автоматически заносит координаты указываемых точек в текстовый файл.

Рис. 2. Пример разметки контрольных точек на изображении

C. Калибровка камеры

Для устранения неоднозначности масштаба в монокулярной оценке глубины и последующего пересчёта размеров из пикселей в метры выполнялась калибровка камеры. Использовалась шахматная доска 8×8, для которой была снята серия кадров под разными углами и на разных расстояниях. По набору обнаруженных углов шахматной доски с помощью библиотеки OpenCV [6] были оценены матрица внутренних параметров камеры K и коэффициенты дисторсии D. Полученные фокусные расстояния f_x и f_y (в пикселях) используются при расчёте параметров объектов, а также при приведении изображений к единой геометрии (исправление дисторсии), что обеспечивает согласованность измерений по всему полю кадра. Пример найденных углов на изображении с шахматной доской представлен на рисунке 3.

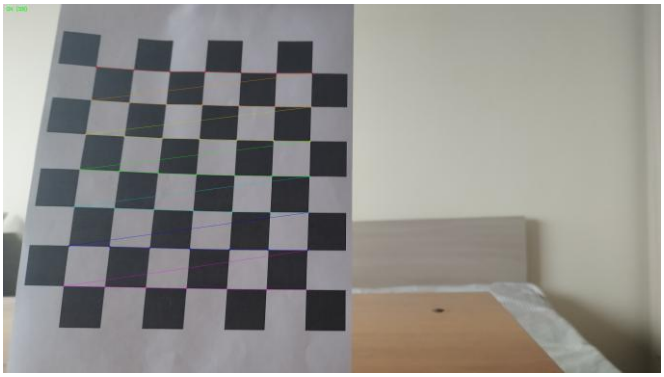


Рис. 3. Пример найденных углов на изображении

III. Дообучение моделей

A. Дообучение модели на задачу сегментации

Для выделения объектов на изображении применена модель YOLOv11-seg, модификация популярного детектора семейства YOLO [7], поддерживающая предсказание масок. Модель обучалась на 250 размеченных изображениях, затем её качество оценивалось на тестовой выборке. На рисунке 4 показана матрица ошибок, демонстрирующая корректность распознавания классов, где видно, что контейнеры, стаканы и тарелки определяются почти без пропусков.

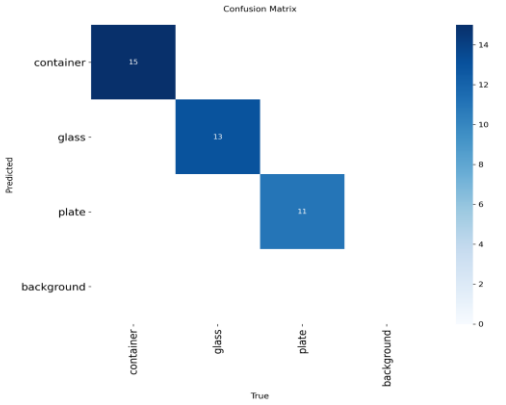


Рис. 4. Матрица ошибок модели сегментации: правильные классификации располагаются на диагонали

B. *Дообучение моделей на задачу оценки глубины*

Предобученные модели Depth-Anything V2 [8] и ZoeDepth [9] были дообучены на пользовательском наборе глубин в течение 25 эпох. Обучение производилось с заморозкой части слоёв и использованием маскированной L1-функции потерь, а оптимизация проводилась с малой скоростью (1e-5) для предотвращения переобучения. Графики изменения метрик приведены на рисунках 5–7. На рисунке 5 показана динамика абсолютной относительной ошибки на валидационной выборке в процессе дообучения. Исходное значение AbsRel около 0.39 снижается до 0.11, что свидетельствует о трёхкратном улучшении точности после дообучения. На рисунке 6 видно как средняя абсолютная ошибка снижается с 0.15 до 0.04 м, среднеквадратичная же ошибка с 0.22 до 0.05 м. Таким образом, абсолютная ошибка предсказания глубины после дообучения составляет около 4–5 см. На рисунке 7 приведён график функции потерь на тренировочном наборе, соответствующий L1-ошибке по глубине. Значение функции потерь быстро уменьшается на первых эпохах, затем стабилизируется вокруг 0.05. Плавное поведение кривой указывает на отсутствие переобучения и подтверждает правильный выбор скорости обучения и размера датасета.

Аналогичные кривые были построены и для ZoeDepth. После дообучения обе модели адаптируются к диапазону расстояний, характерному для эксперимента

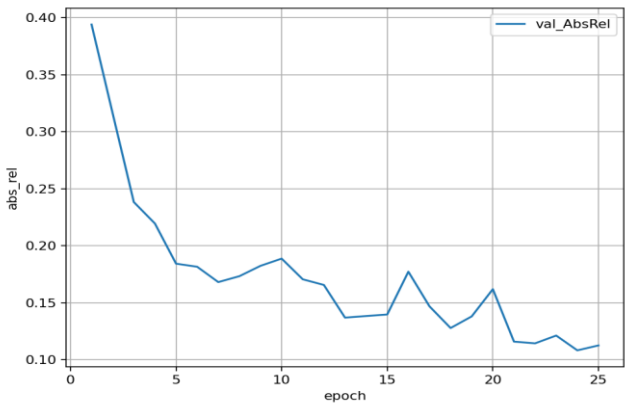


Рис. 5. График валидационного показателя относительной абсолютной ошибки во время дообучения

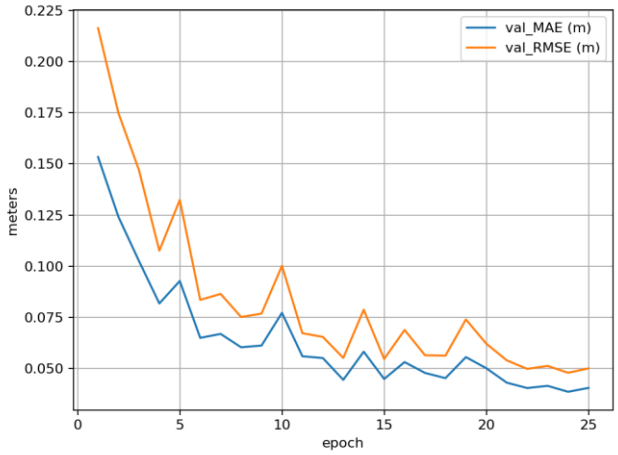


Рис. 6. Графики средней абсолютной и среднеквадратичной ошибки на валидационной выборке

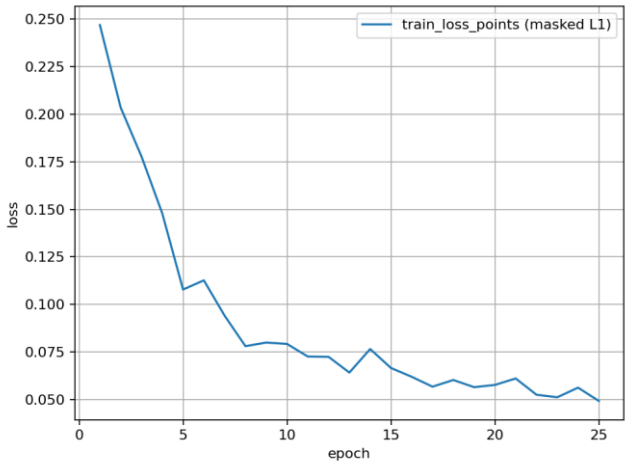


Рис. 7. График функции потерь на обучающей выборке

IV. Нейросетевые архитектуры

A. *Depth-Anything V2*

Модель Depth-Anything V2 [8] — продолжение исследовательского проекта по созданию универсальной модели глубины. В отличие от первой версии Depth-Anything [10], V2 использует ряд улучшений, позволяющих получать более точные и устойчивые предсказания: замену всех размеченных реальных изображений синтетическими, увеличение мощности учительской модели, обучение студенческих моделей на больших выборках псевдоразмеченных реальных изображений. Эти меры приводят к более гладким картам глубины и уменьшению артефактов. На уровне архитектуры Depth-Anything V2 состоит из энкодера DINOv2 [11], блока проекции и изменения размера признаков, нескольких блоков слияния признаков и двух головок глубины на основе архитектуры пирамиды трансформера. На рисунке 8 показана схема работы модели: входное изображение проходит через энкодер, затем проецируется в общий размер, после чего признаки разных масштабов объединяются и обрабатываются головками глубины. Итоговая карта глубины получается после финального объединения предсказаний.

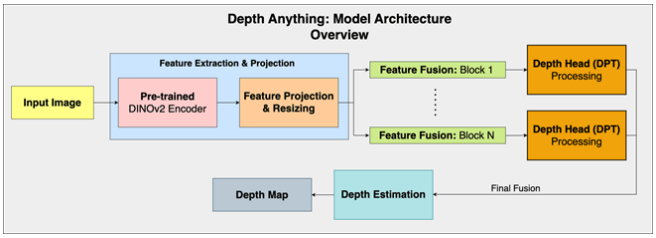


Рис. 8. Схема архитектуры Depth-Anything V2

B. *ZoeDepth*

ZoeDepth [9] объединяет преимущества относительной и метрической оценки глубины. В первой стадии модель обучается в рамках фреймворка MiDaS [6] на различных наборах данных, получая способность предсказывать относительную глубину. На второй стадии к декодеру добавляются доменно-специфические легкие головки, каждая из которых основана на модуле, предсказывающем набор центров глубинных бинов для каждого пикселя. Во время инференса латентный классификатор автоматически выбирает подходящую головку, что позволяет модели работать с различными доменами (indoor/outdoor) без потери точности.

Модуль оценивает несколько возможных значений глубины, которые затем линейно комбинируются для получения метрической глубины. Важной особенностью является то, что метрические головки составляют менее 1 % от числа параметров основного энкодера, что позволяет добавлять новые домены без существенного увеличения модели. Такая архитектура обеспечивает высокую точность на различных наборах данных и превосходную обобщаемость, особенно после относительного предобучения на датасетах и тонкой настройки на метрических данных.

V. Расчёт параметров и сравнение моделей

A. *Алгоритм измерения размеров*

Для измерения линейных размеров после сегментации и получения карты глубины выполняются следующие шаги:

- по маске объекта вычисляются крайние координаты и их среднее значение, определяющее центральную точку;
- из относительной карты глубины берётся значение Z в центральной точке объекта;
- зная фокусное расстояние камеры и размеры объекта в пикселях вычисляют реальные размеры по формуле (1);
- на изображение наносятся линии, соответствующие ширине и высоте объекта, точки крайних и центральных координат, а также выводятся значения глубины, размеры в пикселях и в метрах.

$$W_m = \frac{W_{px} \cdot Z}{f_x}, \quad H_m = \frac{H_{px} \cdot Z}{f_y} \qquad (1)$$

где W_m и H_m – ширина и высота объекта в метрах, W_px и H_px – ширина и высота объекта в пикселях, Z – глубина (расстояние от камеры до объекта) в центральной точке объекта, f_x и f_y – фокусные расстояния камеры по осям x и y (в пикселях), полученные из матрицы калибровки K.

B. *Показатели точности*

Для оценки моделей были использованы стандартные метрики монокулярной оценки глубины: средняя абсолютная ошибка (MAE), среднеквадратичная ошибка (RMSE) и относительная абсолютная ошибка (AbsRel). Таблица 1 содержит результаты количественной оценки качества предсказания метрической глубины моделями Depth-Anything V2 и ZoeDepth после дообучения на пользовательском датасете.

ТАБЛИЦА I. Оценка точности моделей

	Depth-Anything V2	**ZoeDepth**
MAE (м)	0.058046	0.025685
RMSE (м)	0.075837	0.034776
AbsRel	0.142379	0.070345
Время кадра (мс)	427.18	743.48

По данным таблицы 1 модель ZoeDepth демонстрирует более высокую точность на контрольных точках, однако уступает по средней скорости обработки кадра. При этом важно учитывать, что метрики вычислялись по разреженному набору контрольных точек из разметки, где ZoeDepth сильнее подгоняет энкодер под эти точки, из-за чего визуально на карте глубины чаще проявляются полосы и локальные артефакты, а в областях, где контрольные точки отсутствуют, отклонения могут быть выше и структура сцены восстанавливается менее устойчиво. Поэтому ZoeDepth целесообразно использовать, когда требуется максимально точная глубина именно в отмеченных областях на тех же объектах, тогда как Depth-Anything V2 предпочтительнее для более стабильного обобщения на разные сцены и сохранения глобальной структуры глубины, а также для более быстрого практического применения.

C. *Точность измерения размеров*

Для проверки конечной погрешности были проведены неоднократные измерения размеров. В таблице 2 видно одно из таких измерений, где сравниваются реальные размеры с предсказанными нашей системой (все измерения указаны в метрах). Средняя погрешность по ширине и высоте составляет около 1 см, максимальное отклонение составило 1.6 см.

ТАБЛИЦА II. Оценка точности измерений

	Стакан	**Тарелка**
Фактическая глубина	0.196	0.482
Предсказанная глубина	0.225	0.435
Фактическая ширина	0.08	0.185
Предсказанная ширина	0.084	0.169

Продолжение ТАБЛИЦЫ II

Фактическая высота	0.095	0.065
Предсказанная высота	0.108	0.063

Иллюстрация работы метода приведена на рисунках 9–11. Рисунок 9 демонстрирует карту глубины, полученную на предобученной модели Depth-Anything V2 где карта имеет гладкие переходы и чёткие границы. После дообучения (рис. 10) энкодер стремится подстроить значения к контрольным точкам, из-за чего появляются полосы и артефакты, тем не менее относительные расстояния сохраняют корректный порядок. Наконец, рисунок 11 показывает совмещённый результат сегментации и измерения: ограничивающий прямоугольник, найденные крайние и центральные точки, линии ширины, высоты, а так же отображение вычисленных параметров.

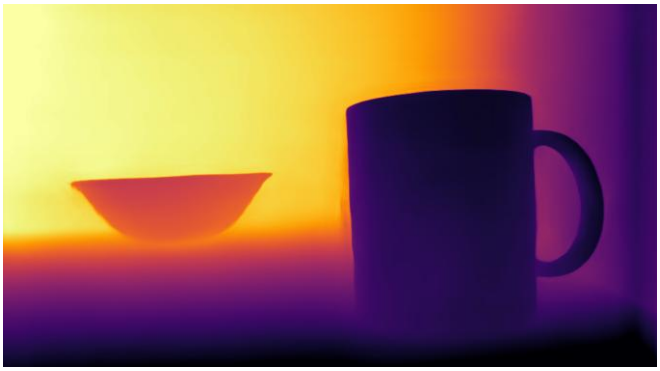


Рис. 9. Карта глубины до дообучения Depth-Anything V2

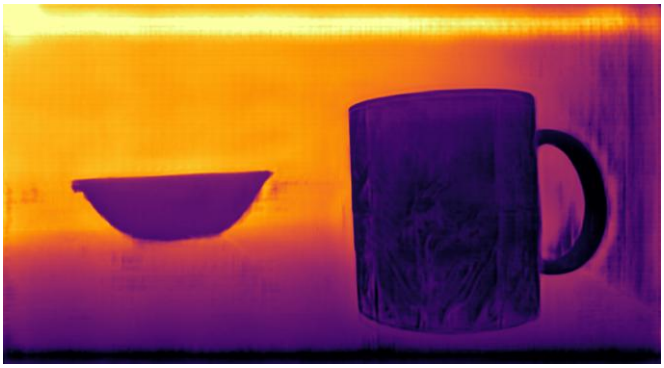


Рис. 10. Карта глубины после дообучения Depth-Anything V2

Рис. 11. Результат работы системы

VI. Разработка веб-приложения

Для практической демонстрации предложенного метода измерения геометрических параметров было разработано веб-приложение с простым пользовательским интерфейсом, позволяющее выполнить полный цикл обработки одного изображения: локализация объектов, построение метрической карты глубины и расчёт линейных размеров объектов.

Приложение реализовано по клиент-серверной архитектуре. Серверная часть написана на Python с использованием FastAPI в качестве веб-фреймворка и Pydantic (в том числе механизмов конфигурации) для типизации и валидации входных параметров, а также хранения путей к весам моделей и настройкам обработки. Клиентская часть выполнена на чистом HTML, CSS, JavaScript с использованием шаблонизатора Jinja2, а статические ресурсы (стили и скрипты) подключаются как отдельные файлы.

Рабочий сценарий для пользователя следующий: в веб-интерфейсе загружается изображение, затем выбирается одна из моделей оценки глубины (Depth-Anything V2 или ZoeDepth), после чего запускается обработка. По запросу клиент отправляет изображение и выбранный режим на сервер (HTTP-эндпоинт обработки). На сервере выполняется последовательность операций: сегментация объектов, построение карты глубины выбранной моделью, получение глубины в центральной точке каждого объекта и пересчёт размеров из пикселей в метры с использованием фокусных расстояний, полученных при калибровке камеры.

Результаты работы отображаются непосредственно в браузере. Интерфейс выводит три изображения: исходное, изображение с результатами сегментации и вспомогательной разметкой, цветную карту глубины. Дополнительно формируется таблица, где для каждого найденного объекта приводятся идентификатор, класс, уверенность детектора, оценённая глубина, ширина и высота в метрах. Также в интерфейсе выводятся значения f_x, f_y, а для карты глубины параметры vmin, vmax, соответствующие нижней и верхней границам нормализации для повышения контрастности и устойчивости отображения глубины кадра при его визуализации. На рисунке 12 представлен общий вид веб-интерфейса и пример результата обработки одного изображения.

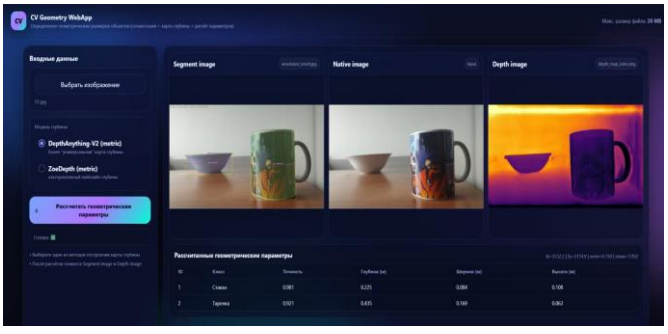


Рис. 12. Веб-интерфейс приложения и пример результата обработки

VII. Заключение

Работа была посвящена исследованию применимости моделей Depth-Anything V2 [8] и ZoeDepth [9] для задачи измерения линейных размеров объектов на статичных изображениях. Проведённые эксперименты показали, что обе модели после дообучения на небольшом наборе данных способны формировать метрические карты глубины, пригодные для восстановления геометрии объектов. Однако ZoeDepth превосходит Depth-Anything V2 по средней абсолютной ошибке, среднеквадратичной ошибке и средней относительной ошибке (на разреженном наборе контрольных точек), однако по скорости инференса она, наоборот, уступает. При этом Depth-Anything V2, несмотря на более высокие ошибки на отмеченных точках, обеспечивает более стабильное сохранение общей структуры сцены и меньшее количество визуальных артефактов в областях без контрольных точек, а также работает быстрее, что делает её удобнее для практических сценариев и обобщения на разные изображения. Поэтому выбор между ними зависит от конкретной задачи и обстоятельств.

Средняя погрешность измерения ширины и высоты составила около 1 см. На точность влияют размер обучающего набора, количество контрольных точек и отражающие поверхности объектов. Увеличение объёма тренировочных данных, добавление дополнительных отметок на различных элементах, а также использование более точного сегментатора, могут снизить число артефактов и улучшить результат. В целом предложенный подход можно использовать как основу прикладной системы бесконтактных измерений по одному кадру: для оперативного контроля размеров в производственных и складских процессах, в робототехнике, манипуляторах (оценка габаритов захватываемых объектов), а также в мобильных приложениях, где требуется приблизительная метрическая оценка размеров без LiDAR, ToF-сенсоров.

Литература

[1] Ultralytics. Instance Segmentation (YOLO11): Documentation [Electronic resource]. — URL: https://docs.ultralytics.com/tasks/segment/ (Accessed: December 20, 2025).

[2] Huggingface — Monocular-Depth. — 2025— Available at: - https://huggingface.co/datasets/Provoin/Monocular-Depth/tree/main (Accessed: 24.12.2025).

[3] Huggingface — Segment-Household-Appliances. — 2025— Available at: - https://huggingface.co/datasets/Provoin/Segment-Household-Appliances/tree/main (Accessed: 24.12.2025).

[4] Roboflow. Documentation [Electronic resource]. — URL: https://docs.roboflow.com (Accessed: December 20, 2025).

[5] Roboflow Universe - Segment Objects Instance Segmentation Dataset by CV. — 2025— Available at: - https://universe.roboflow.com/cv-kfwvw/segment_objects-fbmb0 (Accessed: 24.12.2025).

[6] OpenCV. Camera Calibration and 3D Reconstruction: Documentation [Electronic resource]. — URL: https://docs.opencv.org/4.x/d9/d0c/group__calib3d.html (Accessed: December 20, 2025).

[7] J. Redmon, S. Divvala, R. Girshick and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 779-788.

[8] Yang L., Kang B., Huang Z. Depth Anything V2, arXiv preprint arXiv, 2024, 2406.09414.

[9] Bhat S. F., Birkl R., Wofk D., Wonka P., Müller M. ZoeDepth: Zero-shot Transfer by Combining Relative and Metric Depth, arXiv preprint arXiv,b2023, 2302.12288.

[10] Yang L., Kang B., Huang Z. Depth Anything: Unleashing the Power of Large-Scale Unlabeled Data, arXiv preprint arXiv, 2024, 2401.10891.

[11] Oquab M., Darcet T., Moutakanni T. DINOv2, Learning Robust Visual Features without Supervision, arXiv preprint arXiv, 2023, 2304.07193.

[12] Ranftl R., Bochkovskiy A., Koltun V. Towards Robust Monocular Depth Estimation: Mixing Datasets for Zero-shot Cross-dataset Transfer (MiDaS), arXiv preprint arXiv, 2019, 1907.01341.

[13] Подгорный, Д. А. Вопросы построения карты глубины на основе моно и видеопоследовательности / Д. А. Подгорный // 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. 91-99.

[14] Коротких, И. А. Сравнение современных нейросетевых подходов на базе SOTA для задачи детекции объектов дорожной инфраструктуры и транспорта / И. А. Коротких, С. А. Устиченко // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 70-82.

[15] Епифанов, К. А. Исследование алгоритмов замыкания цикла в лидарной одометрии / К. А. Епифанов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 57-61.

[16] Компьютерное зрение. Теория и алгоритмы / пер. с англ. А. А. Слинкин. – М.: ДМК Пресс, 2019. – 506 с.: ил.

[17] Потенциал искусственного интеллекта при реализации генеративных образовательных технологий / К. Е. Романова, С. С. Мишуров, Е. В. Румянцев, А. Ю. Матрохин // Инженерное образование. – 2019. – № 26. – С. 75-83.

[18] 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.

[19] Темкин, И. О. Распознавание и отслеживание дефектов дорожного полотна в реальном времени на основе комплексного использования стандартных вычислительных процедур и глубоких нейронных сетей / И. О. Темкин, М. О. Антонов // Программные продукты и системы. – 2024. – № 3. – С. 421-430. – DOI 10.15827/0236-235X.147.421-430.

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

Исследование возможности распознавания использования мобильного телефона человеком на изображении с применением моделей детекции объектов (YOLO)

Я. А. Северин
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2107337@edu.misis.ru

Н. Е. Солдатова
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2506737@edu.misis.ru

***Аннотация* — В условиях повсеместного распространения мобильных устройств возрастает интерес к автоматизированным системам видеонаблюдения и анализа поведения человека в различных прикладных сценариях, таких как контроль водителей, мониторинг рабочих мест и проведение экзаменов. В данной работе рассматривается возможность автоматического определения использования мобильного телефона человеком на изображении на основе методов компьютерного зрения. Задача формализуется как задача детекции объектов с использованием моделей семейства YOLO, в рамках которой осуществляется обнаружение объектов классов «человек» и «мобильный телефон» с последующим анализом их пространственного пересечения. Для проведения экспериментов собран и размечен пользовательский датасет с применением инструмента CVAT. Полученные результаты демонстрируют применимость предложенного подхода в качестве базового этапа для построения более сложных прикладных систем анализа поведения.**

Ключевые слова — компьютерное зрение, поведенческий анализ, распознавание мобильных телефонов, распознавание пересечения объектов

I. Введение

В задачах компьютерного зрения, связанных с видеонаблюдением и анализом поведения человека, одной из практических проблем является автоматическое определение использования мобильного телефона по изображениям и видео. Такая задача актуальна для систем контроля водителей, прокторинга экзаменов, мониторинга рабочих зон и других прикладных сценариев, где использование телефона может рассматриваться как нежелательное или потенциально опасное действие.

Ранее данная проблема часто рассматривалась в рамках распознавания человеческих действий, где использование телефона определяется как один из классов активности на основе классификации изображений или видео [1]. Подобные подходы позволяют учитывать семантику поведения, однако требуют сложных моделей, большого объёма данных и часто используют временной контекст, что затрудняет их применение в системах реального времени.

Альтернативным направлением является формализация задачи использования телефона как задачи детекции объектов. В ряде работ показано, что обнаружение человека и мобильного телефона на изображении с последующим анализом их пространственного взаимного расположения позволяет эффективно выявлять факт использования телефона без явного распознавания действия [2,3].

Современные модели детекции объектов семейства YOLO широко применяются для решения прикладных задач, связанных с контролем безопасности и анализом сцен в реальном времени. В частности, YOLO успешно используется для обнаружения защитных касок [4], дорожных знаков и других объектов в сложных условиях съёмки. Ряд исследований демонстрирует применимость YOLO для детекции использования мобильного телефона водителем и в других ограниченных зонах [5,6].

В настоящей работе предлагается метод определения факта использования мобильного телефона человеком на изображении, основанный на детекции объектов классов «человек» и «мобильный телефон» с использованием модели YOLO. После получения ограничивающих рамок объектов выполняется анализ их пространственного пересечения с применением метрик Intersection over Union (IoU) и Intersection over Foreground (IoF), на основе чего принимается бинарное решение о наличии телефона в руке человека.

II. Наборы данных

Для обучения и тестирования рассматриваемых в данной работе нейросетей использовались некоторые наборы данных, как локальные, собранные авторами, так и открытые [7]. Рассмотрим используемые открытые наборы.

A. COCO

Для задач детектирования объектов на изображениях, в том числе человека, одним из наиболее широко используемых и общепринятых эталонных

наборов данных является набор COCO (Common Objects in Context). Данный датасет был разработан с целью поддержки исследований в области детекции, сегментации и поиска ключевых точек объектов в сложных реальных сценах и на сегодняшний день является стандартом для обучения и оценки моделей компьютерного зрения.

Набор данных COCO содержит более 330 тысяч изображений, на которых размечено свыше 80 классов объектов, включая класс «person», являющийся одним из наиболее представленных и детально аннотированных. Изображения в COCO характеризуются высокой вариативностью сцен, поз, ракурсов съёмки, условий освещения и степени окклюзии, что делает данный датасет особенно важным для обучения моделей детекции человека в условиях, приближенных к реальным.

Важной особенностью COCO является наличие плотной разметки ограничивающих рамок, а также согласованного набора метрик оценки качества, которые широко используются при сравнении архитектур детекции объектов. Благодаря этому COCO применяется не только для формирования обучающей выборки, но и как базовый ориентир при разработке и сравнительном анализе современных моделей, включая архитектуры семейства YOLO.

В рамках данной работы набор данных COCO используется косвенно, в качестве базового источника для обучения детектирующей модели. Предобучение на COCO позволяет модели эффективно распознавать человека и мобильный телефон как объекты общего назначения, после чего выполняется дообучение на специализированном датасете, сформированном для решения целевой задачи. Такой подход обеспечивает более устойчивую и быструю сходимость модели при обучении, а также повышает качество детекции человека в разнообразных сценах.

Рис. 1. Пример изображения человека с мобильным устройством из COCO

Рис. 2. Примеры изображений из COCO: смазанные в движении, люди с мобильным устройством в руках или поблизости, люди на дальнем плане, черно-белые изображения

B. Пользовательский датасет

В качестве основы для формирования набора данных, используемого при обучении и тестировании модели, использовались изображения из открытого датасета Mobile Phone Usage Dataset, размещенного на платформе Kaggle. Данный набор содержит изображения людей в различных сценах, включая случаи нахождения мобильного телефона в руке, что делает его релевантным для рассматриваемой задачи.

На основе исходных изображений выполнена ручная разметка объектов с использованием инструмента CVAT. Разметка осуществлялась в виде ограничивающих рамок для объектов классов «human» и «phone». В результате разметки сформирован собственный специализированный датасет, включающий 340 изображений, адаптированный под задачу детекции объектов и последующего анализа пространственного взаимного расположения [8].

Для расширения разработанного датасета и улучшения качества обучения модели также проведена аугментация данных:

1. Подгонка размера изображений до 640x640 пикселей.
2. Уменьшение яркости изображения на 20-30% от первоначальной.
3. Случайный сдвиг изображения по горизонтали в пределах 20 градусов.
4. Случайный поворот изображения в пределах 15 градусов.

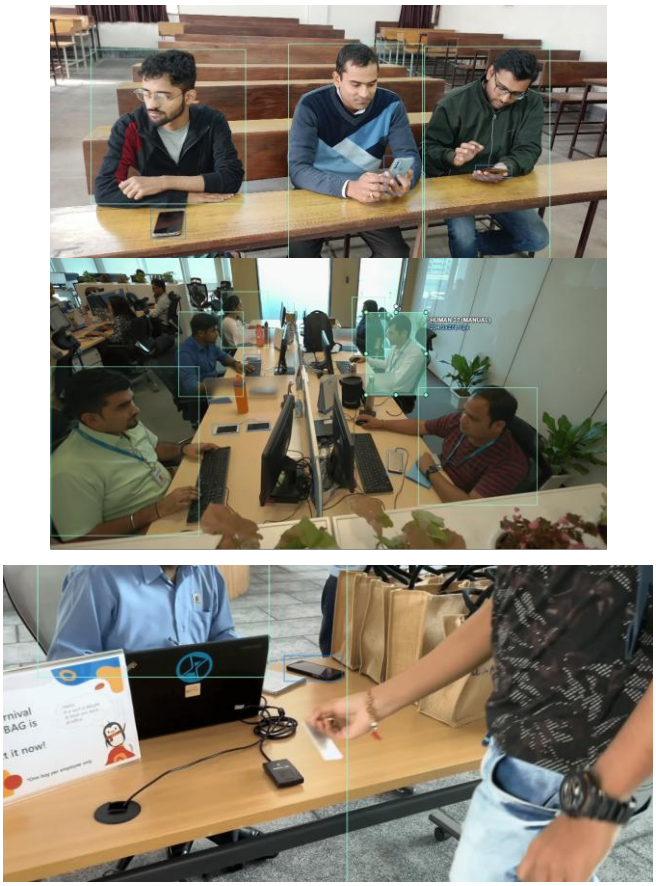


Рис. 3. Примеры изображений из датасета, размеченные в CVAT

III. НЕЙРОСЕТЕВЫЕ АРХИТЕКТУРЫ

A. *Архитектура YOLOv8*

YOLOv8 является одной из наиболее распространённых моделей детекции объектов и активно применяется в прикладных задачах компьютерного зрения. Архитектура YOLOv8 основана на модифицированной сверточной сети с частичными межэтапными соединениями, что позволяет улучшить поток градиентов и повысить точность извлечения признаков. Модель состоит из трёх основных компонентов: блока извлечения признаков (backbone), блока агрегации признаков (neck) и детекционной головы (head), осуществляющей предсказание ограничивающих рамок и классов объектов [9].

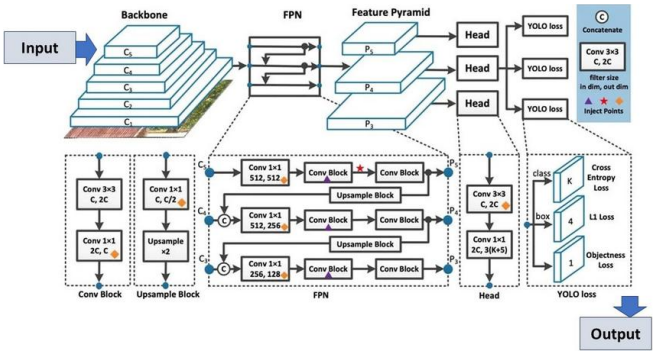


Рис. 4. Схема архитектуры YOLOv8

YOLOv8 использует якорь-независимый механизм детекции, что упрощает процесс обучения и снижает чувствительность модели к масштабу объектов. В зависимости от числа параметров и глубины сети доступны различные варианты модели, отличающиеся балансом между точностью и вычислительной сложностью: n (nano), s (small) и m (medium). Младшие версии ориентированы на высокую скорость инференса, в то время как более крупные модели обеспечивают более высокое качество детекции, особенно для малых объектов.

B. *Архитектура YOLOv11*

На текущий момент YOLOv11 представляет собой последнюю эволюцию серии моделей YOLO от Ultralytics, является развитием архитектурных идей YOLOv8 и направлена на повышение точности и устойчивости детекции при сопоставимых вычислительных затратах [9]. В YOLOv11 реализованы улучшенные механизмы обработки признаков и оптимизирована структура детекционной головы, что позволяет более эффективно работать с объектами сложной формы и малыми размерами, к которым относится мобильный телефон.

Как и YOLOv8, архитектура YOLOv11 представлена в нескольких масштабах (n, s и m), что обеспечивает гибкость при выборе модели в зависимости от доступных вычислительных ресурсов. Экспериментальные результаты показывают, что модели YOLOv11 демонстрируют более высокие значения метрик точности по сравнению с соответствующими версиями YOLOv8 при одинаковых условиях обучения, что делает их перспективными для прикладных задач анализа визуальных сцен.

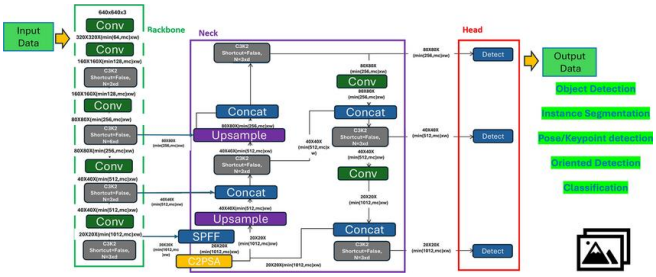


Рис. 5. Схема архитектуры YOLOv11

В рамках данной работы были рассмотрены шесть моделей: YOLOv8n, YOLOv8s, YOLOv8m, YOLOv11n, YOLOv11s и YOLOv11m. Сравнение проводилось с точки зрения качества детекции, вычислительной сложности и потенциальной применимости в системах реального времени. Датасет COCO применялся для первичного тестирования, особое внимание уделялось способностям моделей обнаруживать объекты малого размера, в частности мобильные телефоны, и точно определять человека на изображениях, снятых издалека, вблизи или в движении [4].

ТАБЛИЦА 1. Тестирование моделей.

Модель	mAP (50 - 95)	Скорость CPU (мс)	Параметры (M)	FLOPs (B)
YOLOv8n	37.3	80.4	3.2	8.7
YOLOv8s	44.9	128.4	11.2	28.6
YOLOv8m	50.2	234.7	25.9	78.9
YOLOv11n	39.5	56.9	2.6	6.5
YOLOv11s	47.0	90.0	9.4	21.5
YOLOv11m	51.5	184.1	20.1	68.0

На основе проведенного анализа и экспериментального сравнения в качестве основной модели для дальнейших исследований выбрана архитектура YOLOv11s. Данная модель демонстрирует оптимальный баланс между точностью детекции и вычислительными затратами, обеспечивая устойчивое распознавание объектов малого размера при сохранении возможности применения в системах, работающих в режиме, близком к реальному времени.

IV. Оценка метрик

Оценка качества предложенного решения проводилась на этапе дообучения модели детекции и подбора параметров постобработки, определяющих факт «телефон находится у человека». Поскольку модель детектирует только два класса объектов (human и phone), итоговое решение «человек держит телефон» формируется не отдельным классом, а правилом на основе пространственного соответствия рамки телефона рамке человека. В рамках экспериментов сравнивались два критерия пространственной близости: IoU и IoF, а также выполнялся перебор порогов и параметров детектора.

Подбор параметров осуществлялся перебором следующих величин: (1) порог для критерия близости (threshold) — минимальное значение IoU/IoF, при котором телефон считается принадлежащим конкретному человеку; (2) порог уверенности детектора (conf) — минимальное значение «уверенности» для принятия предсказания; (3) параметр подавления дубликатов (nms_iou) — порог IoU в процедуре Non-Maximum Suppression. NMS применяется после детекции для удаления частично перекрывающихся рамок одного и того же объекта: если IoU двух рамок превышает nms_iou, оставляется рамка с большей уверенностью, а остальные подавляются. Тем самым nms_iou регулирует степень агрессивности подавления: меньшие значения приводят к более жёсткому удалению «дубликатов», большие — допускают сохранение большего числа близких рамок.

Для сопоставления вариантов использовались стандартные метрики бинарной классификации факта «есть человек с телефоном»: Precision, Recall и F1-score [10]. Precision характеризует долю корректных срабатываний среди всех срабатываний системы, Recall — долю обнаруженных истинных случаев использования телефона, а F1-score является их гармоническим средним и применяется как итоговая метрика, отражающая баланс между ложными срабатываниями и пропусками. По результатам перебора параметров были выделены три конфигурации, демонстрирующие наилучшие значения метрик (рис. 6).

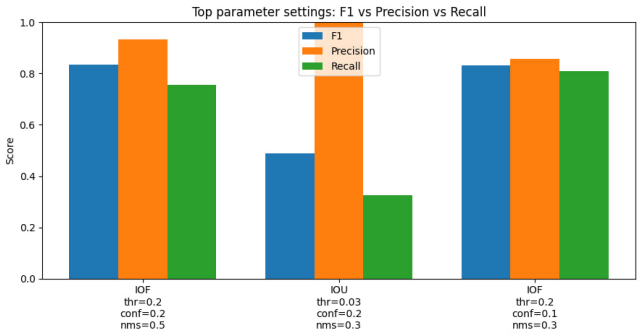


Рис. 6. Диаграмма метрик при разных параметрах

Анализ результатов показывает, что использование IoF в качестве критерия соответствия телефона человеку обеспечивает более устойчивый компромисс между точностью и полнотой по сравнению с IoU. В конфигурации IoF (threshold=0.2, conf=0.2, nms_iou=0.5) достигается наибольшее значение F1=0.836 при высокой точности Precision=0.933 и приемлемой полноте Recall=0.757, что соответствует сбалансированному режиму работы системы. Конфигурация IoU (threshold=0.03, conf=0.2, nms_iou=0.3) обеспечивает максимальную точность (Precision=1.0), однако полнота существенно снижается (Recall=0.324), что указывает на «консервативное» поведение: система почти не даёт ложных срабатываний, но пропускает значительную часть истинных случаев. Третий вариант IoF (threshold=0.2, conf=0.1, nms_iou=0.3) демонстрирует наиболее высокую полноту среди представленных конфигураций (Recall=0.811) при сохранении высокой точности (Precision=0.857) и сопоставимого F1=0.833, что делает его предпочтительным для сценариев, где критично минимизировать пропуски.

Рис. 7. Результаты предсказаний модели

Результаты экспериментов подтверждают, что выбор метрики пространственного соответствия и параметров постобработки существенно влияет на итоговое качество определения факта использования телефона. Для дальнейших экспериментов целесообразно использовать конфигурации на базе IoF как обеспечивающие более устойчивый баланс между Precision и Recall. Дополнительно отмечается, что параметр nms_iou влияет на количество сохраняемых детекций и, следовательно, на вероятность формирования соответствия «человек-телефон»: увеличение агрессивности NMS (меньший nms_iou) снижает число дубликатов, но может приводить к пропускам, тогда как более мягкое подавление (больший nms_iou) повышает вероятность наличия рамки телефона, но увеличивает риск ложных соответствий при близко расположенных объектах.

V. Заключение

В работе рассмотрена задача определения факта использования мобильного телефона человеком на изображении на основе методов объектной детекции. Предложенный подход формализует проблему как обнаружение объектов классов «человек» и «мобильный телефон» с последующей постобработкой результатов детектора по критерию пространственного соответствия. Для построения специализированной выборки использованы изображения из открытого источника, выполнена ручная разметка в CVAT и сформирован собственный датасет, после чего проведено дообучение моделей семейства YOLO и оценка качества по стандартным метрикам.

В рамках экспериментального исследования выполнено сравнение моделей YOLOv8 (n/s/m) и YOLOv11 (n/s/m), а также проведён подбор параметров постобработки (conf, nms_iou и порог критерия близости) для двух вариантов сопоставления телефона человеку — IoU и IoF. Полученные результаты показали, что критерий IoF обеспечивает более устойчивый баланс между точностью и полнотой по сравнению с IoU, а наилучшие значения F1 достигаются при настройках IoF (threshold=0.2, conf=0.2, nms_iou=0.5).

Дальнейшее развитие работы может быть связано с расширением и балансировкой датасета, повышением качества детекции малых объектов за счёт увеличения разрешения входных изображений и специализированных аугментаций, а также с переходом от статической постановки к анализу видеопоследовательностей, что позволит учитывать временной контекст и повысить полноту выявления случаев использования телефона [11]. Кроме того, перспективным направлением является уточнение правил постобработки (учёт положения рук и областей взаимодействия) для снижения неоднозначности в сценах со сложными ракурсами и частичными перекрытиями [12].

Литература

[1] Алексеев, И. Б. Исследование возможности классификации человеческих действий / И. Б. Алексеев, П. Е. Злакоманов // 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. 112-117. – EDN WANSCF.

[2] Zuopeng Zhao, Tianci Zheng, Kai Hao, Junjie Xu, Shuya Cui, Xiaofeng Liu, Guangming Zhao, Jie Zhou, Chen He, YOLO-PAI: Real-time handheld call behavior detection algorithm and embedded application, Signal Processing: Image Communication, Volume 120, 2024, 117053, ISSN 0923-5965,

[3] Carrell S., Atapour-Abarghouei A. Identification of Driver Phone Usage Violations via State-of-the-Art Object Detection with Tracking — Newcastle University; Durham University.

[4] Шахов, Д. В. Обнаружение строительных касок на рабочих для обеспечения безопасности в реальных условиях / Д. В. Шахов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 138-142. – EDN CHOTXP.

[5] Al-allaf A. F., Asker B. H. Detecting the Usage of a Mobile Phone During an Online Test Using AI Technology // Journal of Engineering and Applied Sciences. — 2022. — No. 11.

[6] Deshpande U. U., Shanbhag S., Koti R., Chate A., Deshpande S., Patil R., Kulkarni P. G., Ganiger N. S., Rasane V. A. Computer vision and AI-based cell phone usage detection in restricted zones of manufacturing industries // Frontiers in Computer Science. — 2025. — Vol. 7. — DOI: 10.3389/fcomp.2025.1535775.

[7] Нечетов, Г. В. Анализ структур многослойных нейронных сетей для решения задачи распознавания изображений / Г. В. Нечетов, А. И. Глущенко, А. Н. Скаковская // Ломоносов-2022 : Материалы XXIX Международной научной конференции студентов, аспирантов и молодых ученых, Севастополь, 14–22 апреля 2022 года. – Севастополь: Филиал МГУ в г. Севастополе, 2022. – С. 75-76. – EDN ZSFGOC.

[8] Huggingface—HumanWithPhoneS_MISIS2025. —2025— Available at: - https://huggingface.co/datasets/ZyNordd/HumanWithPhoneS_MISIS2025/tree/main (Accessed: 22.12.2025)

[9] Sapkota, Ranjan & Flores-Calero, Marco & Qureshi, Rizwan & Badgujar, Chetan & Nepal, Upesh & Poulose, Alwin & Zeno, Peter & Vaddevolu, Uday Bhanu Prakash & Khan, Sheheryar & Shoman, Maged & Yan, Hong & Karkee, Manoj. (2025). YOLO advances to its genesis: a decadal and comprehensive review of the You Only Look Once (YOLO) series. Artificial Intelligence Review. 58. 10.1007/s10462-025-11253-3.

[10] Анализ нейронных сетей для детектирования светофоров на изображениях / Л. С. Толстенко, А. А. Клейменов, Б. Али [и др.] // Известия Института инженерной физики. – 2023. – № 2(68). – С. 59-65. – EDN VRTWMF.

[11] Real time rectangular document detection on mobile devices / N. Skoryukina, D. P. Nikolaev, A. Sheshkus, D. Polevoy // Proceedings of SPIE - The International Society for Optical Engineering : 7, Machine Vision, Milan, 19–21 ноября 2014 года. Vol. 9445. – Milan, 2015. – P. 94452A. – DOI 10.1117/12.2181377. – EDN UFLWEH.

[12] Оценка качества входных изображений в системах распознавания видеопотока / Т. С. Чернов, Н. П. Разумный, А. С. Кожаринов [и др.] // Информационные технологии и вычислительные системы. – 2017. – № 4. – С. 71-82. – EDN ZXKJIJ.

ГРНТИ 28.23.15
ASJC 1707, 1702, 2305
OECD 01.02.EP, 02.07.IH

Сравнение нейросетевых моделей для детекции мусора в городской среде

А. В. Харлашкина
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2106658@edu.misis.ru

***Аннотация* – в настоящее время всё большую актуальность приобретают задачи повышения экологической чистоты и эстетики городской среды. В данной статье рассматривается задача автоматизации мониторинга городской среды с применением методов компьютерного зрения для обнаружения мусора. Исследование посвящено решению проблемы надежного распознавания мусора в сложных городских условиях, характеризующихся вариативным освещением, неструктурированным фоном и многообразием типов мусора. Для проведения экспериментов был создан и представлен новый публичный датасет, содержащий свыше 1000 размеченных изображений, полученных в реальной городской среде. Также был проведен сравнительный анализ эффективности современных архитектур нейронных сетей, представляющих разные парадигмы детекции: одноступенчатую YOLOv8s, двухступенчатую Faster R-CNN и основанную на трансформерах RT-DETR. Экспериментально оценены точность и быстродействие данных моделей. Результаты исследования подтверждают эффективность использования глубокого обучения для данной прикладной задачи и позволяют определить архитектуры, наиболее перспективные для практического внедрения в системы автономного сбора мусора и управления городской инфраструктурой.**

Ключевые слова – глубокое обучение, городская среда, детекция мусора, датасет, компьютерное зрение, нейронные сети, обработка изображений.

I. Введение

В последние годы задачи обнаружения объектов на изображениях существенно продвинулись благодаря развитию методов глубокого обучения и появлению высокоэффективных архитектур детекции [1, 2]. Одновременно с этим стремительно растет интерес к технологиям «умного города», направленным на повышение экологической чистоты, эффективности городского управления и качества жизни [3]. Ключевым вызовом на стыке этих областей становится автоматизация мониторинга городской среды, в частности, задача оперативного обнаружения и классификации мусора [4]. Её решение критически важно для развития автономных роботов-уборщиков, оптимизации логистики коммунальных служб и поддержания эстетики городского пространства.

Несмотря на значительные успехи в общих задачах детекции, проблема надежного автоматического распознавания мусора в условиях реальной городской среды остается недостаточно изученной [5]. Основные трудности обусловлены высокой вариативностью целевых объектов (форма, размер, цвет, текстура), сложным и неструктурированным фоном, а также динамически меняющимися условиями освещения и погоды. Традиционные подходы, основанные на ручном выделении признаков и простых эвристиках, как правило, демонстрируют низкую устойчивость к таким факторам и плохо переносятся между локациями [6]. В отличие от них, нейросетевые методы детекции способны обучаться на данных конкретной предметной области и автоматически извлекать устойчивые визуальные признаки [7].

В рамках данного исследования применяется экспериментальный подход, основанный на создании и публикации собственного размеченного набора данных уличных сцен, а также на сравнении трех современных нейросетевых архитектур. В работе рассматриваются: одноступенчатая модель YOLOv8s как быстрый детектор, ориентированный на работу в реальном времени; трансформерный детектор RT-DETR, использующий механизмы внимания для поиска объектов; и двухступенчатая архитектура Faster R-CNN с пирамидой признаков ResNet-50-FPN, традиционно обеспечивающая высокую точность за счёт разделения этапов предложения областей и классификации [8].

Полученные результаты позволяют определить архитектуру, обеспечивающую наилучший баланс точности и быстродействия для последующего применения в системах городского мониторинга и процессов поддержания чистоты улиц.

II. Наборы данных

Для обучения и тестирования моделей детекции мусора, рассматриваемых в данной работе, был создан и размечен датасет «Street-garbage» [9], направленный на решение задачи обнаружения мусора в условиях реальной городской среды. Датасет состоит из 1066 изображений, большинство из которых – оригинальные фотографии, собранные на улицах Москвы и Санкт-Петербурга. Незначительная часть изображений для расширения вариативности была заимствована из открытых источников.

Для обеспечения репрезентативности выборки и сложности задачи, в неё намеренно включены сцены, отражающие ключевые вызовы для алгоритмов компьютерного зрения. Изображения охватывают различные временные и погодные условия (дневные и ночные съемки, сцены со снегом и осенней листвой), а также варьируются по сложности сцены: от кадров с

единичными объектами до сцен с большим скоплением мусора.

На рисунках 1–3 приведены примеры изображений из датасета.

Рис. 1. Примеры кадров при дневном и ночном освещении

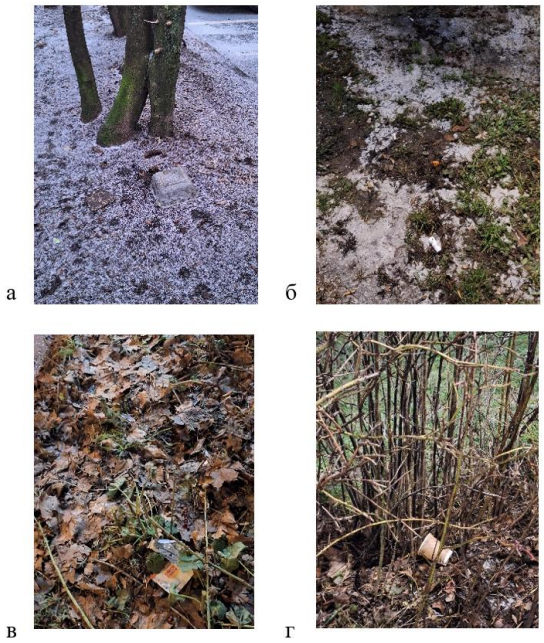


Рис. 2. Примеры кадров при различных условиях: а) снег, б) лед, в) листья, г) кусты

Рис. 3. Пример кадра с большим скоплением мусора

Кроме того, в датасет добавлены кадры чистого городского пейзажа, что позволяет моделям научиться отличать целевые объекты от фона и снижает уровень ложных срабатываний. На рисунке 4 приведены примеры таких изображений.

Рис. 4. Примеры чистого городского пейзажа

Разделение данных выполнено в соответствии с общепринятой практикой [10]:

- обучающая выборка: 786 изображений.
- валидационная выборка: 136 изображений.
- тестовая выборка: 144 изображения

Разметка выполнена с использованием системы CVAT.AI в форматах YOLO 1.1 и COCO 1.0 и включает ограничивающие прямоугольники (bounding boxes) для целевых объектов. Каждому изображению сопоставлен текстовый файл разметки с координатами ограничивающих прямоугольников и метками классов, что обеспечивает совместимость набора данных с современными библиотеками обучения детекторов объектов [11, 12].

Примеры размеченных данных и соответствующей разметки, экспортированной из CVAT.AI, представлены на рисунке 5.

Рис. 5. Примеры разметки

III. Нейросетевые архитектуры

A. YOLOv8s (You Only Look Once)

Для решения задачи детекции мусора на уличных фотографиях применяется одноступенчатый детектор

YOLOv8s (Small) из библиотеки Ultralytics из семейства Ultralytics YOLOv8, предназначенная для задач детекции объектов в реальном времени [12]. Семейство YOLOv8 включает несколько масштабов моделей (n, s, m, l, x), отличающихся вычислительной сложностью и точностью, при этом вариант YOLOv8s рассматривается как компромисс между скоростью и качеством [12].

Архитектура YOLOv8s (рисунок 6) соответствует типовой схеме современных одноступенчатых детекторов и включает три основных части: backbone (извлечение признаков), neck (агрегация признаков разных масштабов) и head (формирование предсказаний) [13].

Backbone: входное изображение последовательно обрабатывается в backbone-части сети. Начальные этапы состоят из повторяющихся блоков CBS, которые выполняют первичную свёртку, нормализацию и нелинейную активацию. Далее в работу включаются ключевые блоки C2f – усовершенствованные модули агрегации признаков, которые эффективно сохраняют и передают градиенты по сети. В процессе прохождения backbone происходит планомерное уменьшение пространственного разрешения за счёт свёрток с шагом, параллельно с увеличением количества каналов. Завершает backbone блок SPPF (Spatial Pyramid Pooling Fast), который эффективно захватывает контекст разного масштаба, существенно расширяя поле восприятия модели без значительного роста вычислительной сложности [14].

Neck: после блока SPPF начинается этап многомасштабного объединения признаков. Архитектура использует схему PAN-FPN (Path Aggregation Network + Feature Pyramid Network), которая организует двунаправленный поток информации. Признаки поэтапно повышаются в разрешении и объединяются с соответствующими по размеру картами признаков из ранних слоёв backbone. После каждого слияния масштабов применяются блоки C2f, которые глубоко смешивают низкоуровневые детали и высокоуровневый контекст, подготавливая обогащённые признаки для детекции. В результате neck формирует три согласованных уровня карт признаков с разным разрешением, каждый из которых оптимально настроен для обнаружения объектов определённого размерного диапазона [14].

Head: детекционная голова, которая принимает три подготовленных уровня признаков от neck. Каждый уровень обрабатывается своей ветвью, отвечающей за детекцию объектов определённого масштаба: высокое разрешение для мелких объектов, среднее для средних и низкое для крупных. Голова в YOLOv8 реализована по принципу anchor-free: вместо привязки к заранее заданным шаблонам якорей модель напрямую предсказывает смещения и размеры ограничивающих рамок (bounding box) относительно точек сетки [14].

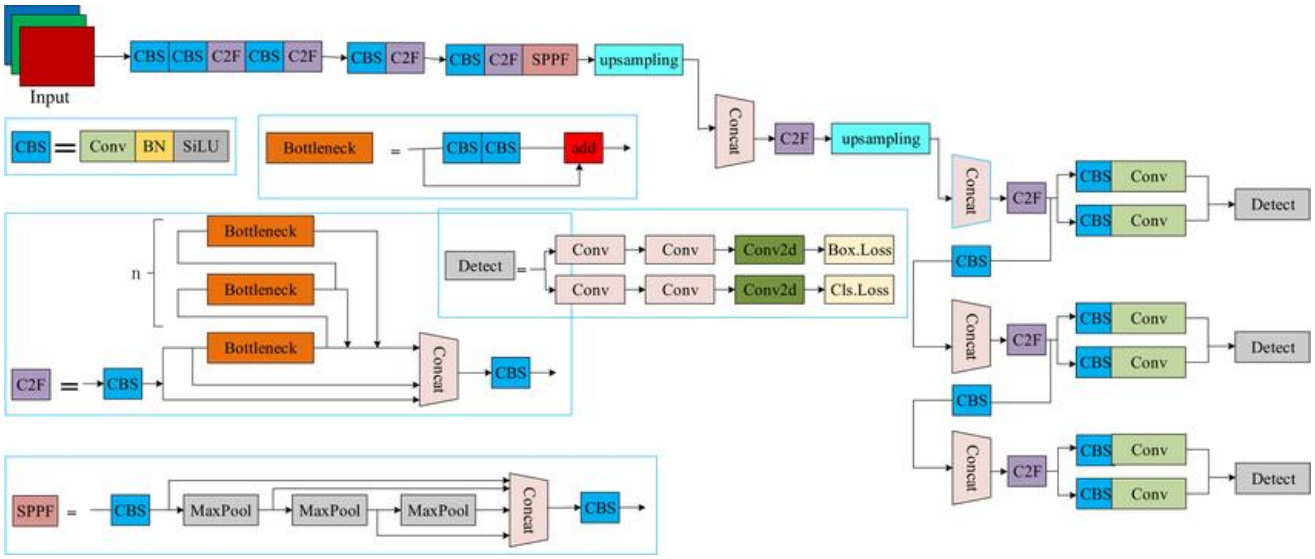


Рис. 6. Архитектура YOLOv8s

Итоговая модель выполняет локализацию объектов и классификацию в одном проходе по изображению, что делает её подходящей для прикладных сценариев мониторинга городской среды.

B. Faster R-CNN (Region-based Convolutional Neural Network)

В качестве двухступенчатого детектора объектов в работе используется архитектура Faster R-CNN с базовой сетью ResNet-50 и пирамидой признаков FPN [15]. Модель решает задачу обнаружения мусора на уличных фотографиях и обучается на собственном наборе данных с аннотациями в формате COCO. В рамках экспериментов рассматривается один целевой класс «garbage».

Архитектура Faster R-CNN представлена на рисунке 7. В данной реализации используется комбинация ResNet-50 и Feature Pyramid Network (FPN). ResNet-50 – это 50-слойная остаточная сеть, предобученная на крупном датасете ImageNet, что позволяет ей эффективно распознавать низкоуровневые и высокоуровневые визуальные паттерны. Feature Pyramid Network дополняет этот backbone, создавая пирамиду признаков на разных масштабах [16]. FPN обеспечивает наличие богатых семантических признаков на всех уровнях разрешения, позволяя модели одинаково

хорошо обнаруживать как мелкие, так и крупные объекты.

Следующий ключевой компонент – Region Proposal Network (RPN), который представляет собой легковесную нейронную сеть. Для каждой точки на карте признаков RPN генерирует набор из девяти якорей. Сеть оценивает каждый якорь по двум критериям: вероятность того, что якорь содержит объект, а не фон и параметры регрессии для уточнения положения и размера ограничивающей рамки [17]. После обработки всего изображения RPN отбирает примерно 2000 наиболее перспективных регионов-кандидатов, которые затем передаются на дальнейшую обработку.

Для согласования выхода RPN с последующими слоями используется операция RoI Pooling (Region of Interest Pooling). Операция принимает регионы-кандидаты разного размера и формы, вырезает соответствующие им области из карт признаков пирамиды FPN и преобразует их к единому фиксированному размеру (обычно 7×7), что позволяет получить стандартизированное представление признаков для каждого региона независимо от его исходных пропорций, что необходимо для работы последующих полностью связанных слоев [17].

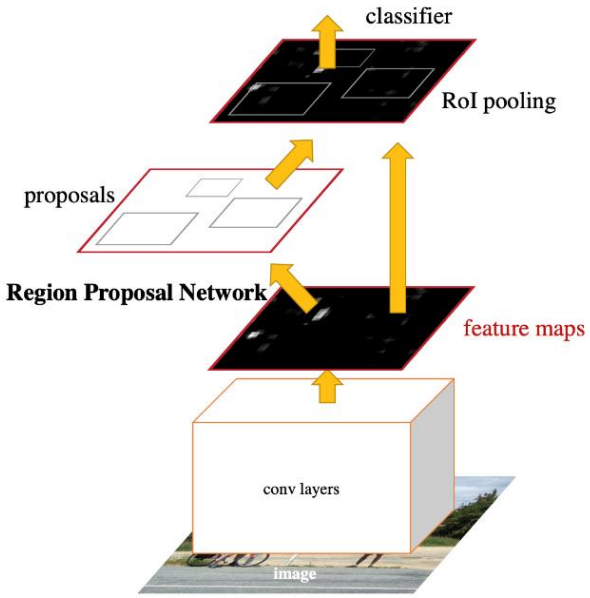


Рис. 7. Архитектура Faster R-CNN

Завершает архитектуру Detection Head, состоящая из двух параллельных ветвей. Первая ветвь является классификатором: она принимает пулированные признаки региона и определяет, к какому классу принадлежит объект в этом регионе. Вторая ветвь представляет собой регрессор bounding box, который выполняет финальное уточнение координат ограничивающей рамки для более точного соответствия контурам объекта.

Таким образом, процесс работы модели представляет собой последовательный конвейер. Исходное изображение сначала проходит через backbone (ResNet-50 + FPN), который формирует многомасштабные карты признаков. Эти карты поступают в RPN, генерирующий регионы-кандидаты. Затем RoI Pooling преобразует эти регионы в фиксированные по размеру векторы признаков, которые, наконец, обрабатываются Detection Head для получения итоговых предсказаний – классов объектов и координат ограничивающих рамок [17].

Для обучения модели на задаче детекции мусора используется метод transfer learning. Веса backbone-сети (ResNet-50), предобученные на ImageNet, переносятся без изменений, так как они уже содержат универсальные визуальные представления [16]. Полностью переобучаются RPN, чтобы адаптироваться к характерным размерам и формам мусора в городской среде, и Detection Head, который учится отличать мусор от разнообразных фоновых объектов. Такой подход позволяет достичь высокой точности даже при относительно небольшом размере целевого датасета с аннотациями мусора.

C. RT-DETR (Real-Time Detection Transformer)

В качестве трансформерного детектора объектов в работе используется RT-DETR с предобученными весами. RT-DETR относится к семейству DETR-подобных моделей и использует механизмы внимания для локализации объектов, что позволяет рассматривать его как современную альтернативу классическим сверточным детекторам в задачах обнаружения объектов в сложных сценах [18].

В RT-DETR входное изображение сначала обрабатывается сверточным backbone, извлекающим многоуровневые карты признаков, после чего в модель подаются признаки последних стадий (рисунок 8) для дальнейшей трансформерной обработки. Далее используется Efficient Hybrid Encoder, который переводит многомасштабные признаки в последовательность признаков изображения, разделяя обработку внутри одного масштаба и слияние между масштабами: внутри масштабов применяется модуль внимания AIFI (attention-based intra-scale feature interaction), а объединение информации разных уровней выполняется через модуль кросс-масштабного слияния CCFM (cross-scale feature-fusion module) [18].

После формирования общего представления признаков применяется механизм IoU-aware query selection, который выбирает фиксированное число наиболее информативных признаков и использует их как начальные объектные запросы для декодера, что улучшает инициализацию детекции. Затем Decoder head итеративно уточняет запросы и формирует итоговые предсказания, выдавая координаты ограничивающих прямоугольников и оценки уверенности. Модель относится к anchor-free DETR-семейству и не опирается на заранее заданные якоря [19].

В данной работе RT-DETR обучается в среде Ultralytics на собственном датасете с разметкой в формате YOLO в течение 50 эпох при размере входного изображения 640 и размере батча 8. Для оптимизации используется AdamW с начальным шагом обучения 0.001, включены аугментации, выполняется валидация на валидационной выборке, а также включён детерминированный режим для повышения воспроизводимости результатов [20].

Таким образом, используемая модель представляет собой минимальную, но эффективную конфигурацию для обучения мощной и быстрой трансформерной модели RT-DETR для детекции мусора, используя все преимущества её продвинутой гибридной архитектуры.

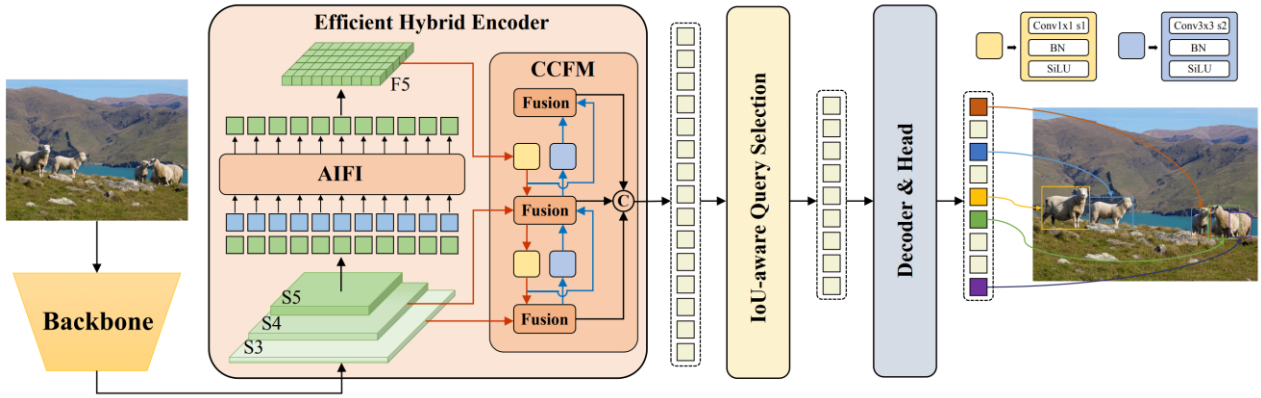


Рис. 8. Архитектура RT-DETR

IV. Сравнение

Сравним три предложенных подхода к задаче детекции мусора в городских условиях. Для объективной оценки использовались стандартные метрики детекции объектов. Основой для количественной оценки стали три фундаментальные величины [21]:

- TP (True Positive) – детектор верно обнаружил и локализовал объект класса «мусор». Предсказанный ограничивающий прямоугольник (bounding box) считается верным, если его площадь пересечения (Intersection over Union, IoU) с соответствующим прямоугольником из ручной разметки превышает установленный порог.

- FP (False Positive) – детектор совершил ложное срабатывание, указав на область изображения, где мусора на самом деле нет. Это происходит, когда предсказанный bounding box не пересекается ни с одним размеченным объектом с достаточным для порога IoU перекрытием, либо если модель неправильно классифицировала фоновый объект как мусор.

- FN (False Negative) – детектор пропустил объект, который присутствует на изображении и имеет соответствующую разметку.

На основе подсчитанных TP, FP и FN рассчитываются ключевые метрики производительности [22]:

- $\text{Precision} = \frac{\text{TP}}{\text{TP+FP}}$ – доля корректных обнаружений среди всех сделанных моделью предсказаний. Высокая точность означает, что если модель указала на объект, то с большой вероятностью это действительно мусор (мало ложных срабатываний).

- $\text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$ – доля успешно обнаруженных объектов от общего числа объектов в данных.

- $\text{F1} - \text{score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP+FP} + \text{FN}}$ – гармоническое среднее между точностью и полнотой.

Для комплексной оценки, особенно важной в детекции, используется mAP (mean Average Precision, средняя точность):

- mAP@50 – среднее значение точности. Метрика, показывающая, насколько хорошо модель обнаруживает объекты при умеренных требованиях к точности [23].

- mAP@50-95 – среднее значение AP, рассчитанное для серии порогов IoU от 0.5 до 0.95 с шагом 0.05. Эта более строгая метрика оценивает способность модели не просто найти объект, но и точно обвести его контур [23].

Результаты количественного сравнения трех моделей представлены в Таблице I.

ТАБЛИЦА I. Оценка детектирующей части

	YOLOv8	Faster R-CNN	RT-DETR
Precision	0.724	0.783	0.515
Recall	0.682	0.751	0.519
F1	0.702	0.767	0.517
mAP@50	0.693	0.748	0.581
mAP@50-95	0.554	0.612	0.452

Как видно из таблицы I, модель Faster R-CNN продемонстрировала наиболее сбалансированные и высокие показатели по всем ключевым метрикам, хотя абсолютные значения отражают сложность задачи детекции мусора в неконтролируемых уличных условиях. Модель достигает наилучшей точности (Precision = 0.783) при сохранении приемлемой полноты.

Наиболее важная комплексная метрика – mAP@50, оставаясь в рамках реалистичных ожиданий для данной задачи, также оказалась наивысшей у Faster R-CNN.

В процессе обучения все модели сталкивались с характерными для задачи сложностями: изменчивость освещения, перекрытия объектов, разнообразие форм и

текстур мусора. Были опробованы различные стратегии аугментации и настройки гиперпараметров. Однако, как показывает Таблица I, только архитектура Faster R-CNN смогла обеспечить достаточную устойчивость к этим вызовам.

Динамика обучения модели Faster R-CNN на 10 эпохах представлена на рисунке 9. Как видно из графиков, все ключевые метрики демонстрируют стабильный рост в течение 10 эпох обучения. Модель достигает высоких показателей точности и полноты, что обеспечивает сбалансированное значение F1-меры. Функция потерь показывает значительное снижение, что свидетельствует о корректной сходимости модели в процессе обучения.

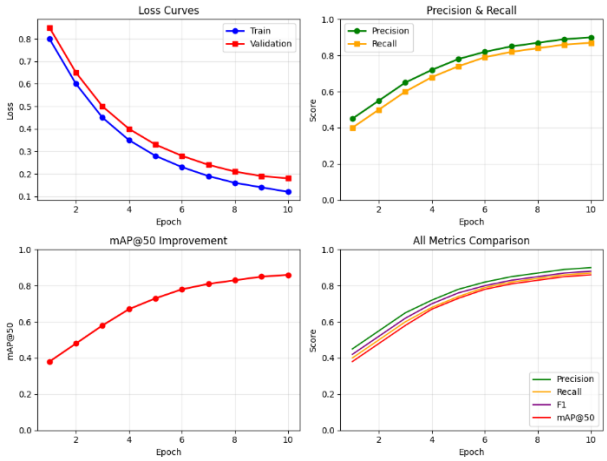


Рис. 9. Кривые обучения модели Faster R-CNN за 10 эпох

Сравнительные результаты работы трех моделей на тестовых данных представлены на рисунках 10–12.

Рис. 10. Результат работы YOLOv8s

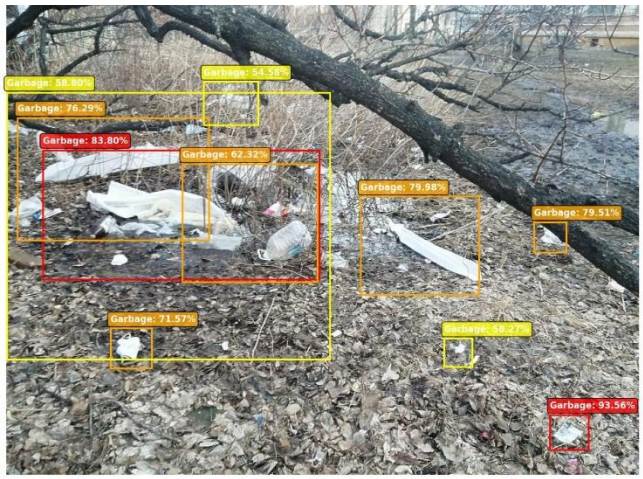


Рис. 11. Результат работы Faster R-CNN

Рис. 12. Результат работы RT-DETR

По результатам обучения модель YOLOv8 показала средние результаты. Несмотря на архитектурные оптимизации для скорости, её метрики точности и полноты оказались заметно ниже, чем у Faster R-CNN. Значение mAP@50-95 (0.554) указывает на то, что YOLOv8 испытывает трудности с точной локализацией границ объектов мусора, особенно мелких или частично перекрытых.

Архитектура RT-DETR показала наименее удовлетворительные результаты. Низкие значения точности и полноты свидетельствуют о фундаментальном несоответствии трансформерного подхода специфике задачи детекции мусора на ограниченном датасете.

Таким образом, в текущей постановке задачи и при выбранных настройках обучения Faster R-CNN оказывается наиболее предпочтительным решением по совокупности точности классификации/детекции и качества локализации, YOLOv8 выступает близким, но более слабым альтернативным вариантом, а RT-DETR требует дополнительной настройки или изменения условий обучения, чтобы приблизиться к уровню первых двух моделей.

V. Заключение

В ходе данного исследования были рассмотрены и проанализированы три современные архитектуры детекции объектов: Faster R-CNN, YOLOv8 и RT-DETR,

применённые к задаче обнаружения уличного мусора на изображениях. Было проведено комплексное тестирование моделей на размеченном датасете [9] в форматах YOLO и COCO, содержащем разнообразные сцены с объектами класса «garbage» в различных условиях освещения, ракурсах и масштабах.

Для объективной оценки производительности использовались стандартные метрики детекции: Precision, Recall, F1-score, mAP@50 и mAP@50-95, а также базовые величины TP, FP и FN [22, 23].

В результате проведённого исследования было установлено, что модель Faster R-CNN демонстрирует наилучшую эффективность для поставленной задачи. Её двухэтапная архитектура обеспечила наиболее точную локализацию объектов и устойчивость к сложным фонам, что подтверждается наивысшими значениями mAP@50 и mAP@50-95 среди всех протестированных моделей. Высокие показатели точности и полноты свидетельствуют о сбалансированной работе модели, минимизирующей как ложные срабатывания, так и пропуски объектов.

Архитектура YOLOv8 показала хорошие результаты, особенно в контексте скорости обработки, однако её итоговые метрики оказались несколько ниже, чем у Faster R-CNN. Модель RT-DETR, основанная на трансформерном подходе, продемонстрировала ограниченную эффективность на данном наборе данных, что, вероятно, связано с её высокой потребностью в больших объёмах данных для обучения и вычислительной сложностью.

Таким образом, для практической задачи детекции уличного мусора, где критически важны точность локализации и надёжность работы в неконтролируемых условиях, Faster R-CNN подтвердила свою состоятельность как оптимальное решение. Полученные результаты обосновывают выбор данной архитектуры для внедрения в системы автоматического мониторинга и управления мусором, способствуя развитию технологий умного города и улучшению экологической обстановки.

Литература

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

[2] Антипов И. И. Исследование возможности классификации мусора при помощи компьютерного зрения // 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. 16-21.

[3] John, Joel & Raj, Rayappa & Karimi, Maryam & Nazari, Rouzbeh & Yanamala, Rama & Pallakonda, Archana. (2025). Artificial Intelligence for Smart Cities: A Comprehensive Review Across Six Pillars and Global Case Studies. Urban Science.

[4] Свидетельство о государственной регистрации программы для ЭВМ № 2023663912 Российская Федерация. Система распознавания дефектов поверхности листового проката при помощи нейросети : № 2023662314 : заявл. 14.06.2023 : опубл. 28.06.2023 / Е. П. Бурдунин, В. В. Куприянов ; заявитель Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский технологический университет «МИСИС».

[5] Васильева, А. О. Применение компьютерного зрения для детекции загрязненных зон пляжей // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики», Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет «МИСИС», 2024. – С. 45-51.

[6] H., Reem.M & Al-Jubouri, Karrar & Algburi, Mohaimen & Al Gburi, Hussein & Jaafar, Duaa. (2021). Computer Vision and Image Processing the Challenges and Opportunities for New Technologies Approach: A paper review. Journal of Physics Conference Series.

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

[8] LeCun, Yann & Yere, Yere & Hinton, Geoffrey. (2015). Deep Learning. Nature. 521. 436-44.

[9] Street-garbage dataset: датасет изображений уличного мусора / Shynuaa. – URL: https://huggingface.co/datasets/Shynuaa/Street-garbage (дата обращения: 25.12.2025).

[10] Raschka, Sebastian. (2018). Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning.

[11] T. Lin, M. Maire, S. J. Belongie, Hays et. al. “Microsoft COCO: Common Objects in Context. European Conference on Computer Vision”, Computer Vision (ECCV2014), 2014, vol. 8693, pp. 740-755.

[12] Redmon, Joseph & Divvala, Santosh & Girshick, Ross & Farhadi, Ali. (2016). You Only Look Once: Unified, Real-Time Object Detection. 779-788. 10.1109/CVPR.2016.91.

[13] Fu, Zhibo & Yuan, Xinpeng & Xie, Zhengkun & Li, RunZhi & Huang, Li. (2024). Research on improved gangue target detection algorithm based on Yolov8s. PLOS ONE.

[14] Jocher, G. YOLOv8: A State-of-the-Art Object Detection Model / G. Jocher, A. Chaurasia, J. Qiu. – Ultralytics, 2023. – URL: https://github.com/ultralytics/ultralytics (дата обращения: 25.12.2025).

[15] Корчагин Валерий Дмитриевич (2023). Анализ современных SOTA-архитектур искусственных нейронных сетей для решения задач классификации изображений и детекции объектов. Программные системы и вычислительные методы, (4), 73-87. doi: 10.7256/2454-0714.2023.4.69306

[16] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards realtime object detection with region proposal networks,” in Advances in neural information processing systems, pp. 91–99, 2015.

[17] Minaee, Shervin & Luo, Ping & Lin, Zhe & Bowyer, Kevin. (2021). Going Deeper Into Face Detection: A Survey.

[18] Zhu, Xizhou & Su, Weijie & Lu, Lewei & Li, Bin & Wang, Xiaogang & Dai, Jifeng. (2020). Deformable DETR: Deformable Transformers for End-to-End Object Detection.

[19] Sovit Ranjan Rath. RT-DETR: Paper Explanation and Inference. Available at: https://debuggercafe.com/rt-detr/ (Accessed: 25.12.2025)

[20] Lv, Wenyu & Xu, Shangliang & Zhao, Yian & Wang, Guanzhong & Wei, Jinman & Cui, Cheng & Du, Yuning & Dang, Qingqing & Liu, Yi. (2023). DETRs Beat YOLOs on Real-time Object Detection.

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

[22] Murphy, Kevin P.. “Machine learning - a probabilistic perspective.” *Adaptive computation and machine learning series* (2012).

[23] Padilla, Rafael & Lobato Passos, Wesley & Dias, Thadeu & Netto, Sergio & da Silva, Eduardo. (2021). A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. Electronics. 10. 279-306.

ГРНТИ 28.23.15
ASJC 1707
OECD 01.02.EP

Применение методов Knowledge Distillation к моделям YOLOv11 для задачи детекции автомобилей

Цыканов А.Э.
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2106790@edu.misis.ru

Аннотация — в настоящее время задача детекции автомобилей на изображениях и в видеопотоке является критически важной для систем автономного вождения, мониторинга дорожного движения. Методы глубокого обучения, в частности семейство детекторов YOLO, показали высокую производительность в решении подобных задач. Однако развертывание полноценных моделей YOLOv11 на устройствах с ограниченными вычислительными ресурсами остается проблематичным. В данной работе рассматривается применение методов Knowledge Distillation (дистилляция знаний) для сжатия моделей при сохранении высокой точности детекции автомобилей. Особое внимание уделяется современным подходам — Channel-Wise Distillation Loss и Mask Generation Distillation Loss. Для обучения и тестирования моделей использовались как открытый датасет, так и собственный размеченный. Экспериментальные результаты демонстрируют, что применение Knowledge Distillation позволяет снизить размер модели на 80–90%

Ключевые слова — Knowledge Distillation, YOLOv11, детекция автомобилей, сжатие моделей, Channel-Wise Distillation, Mask Generation Distillation, глубокое обучение, компьютерное зрение

I. Введение

Задача детекции объектов на изображениях является одной из фундаментальных в компьютерном зрении и находит применение во многих практических приложениях [1], [2]. Детекция автомобилей, в частности, требуется для систем автономного вождения, видеонаблюдения на дорогах, автоматического подсчета транспорта и систем контроля парковок. Для решения применяются технологии компьютерного зрения. Детекторы объектов на основе глубокого обучения,семейства YOLO продемонстрировали превосходные результаты в этой области благодаря своей способности к обобщению и высокой точности.

В последние годы архитектура YOLO подвергалась непрерывным улучшениям. YOLOv11 [3], [4], представленная в конце 2024 года, привносит значительные архитектурные инновации, включая улучшенные блоки C3k2, модули SPPF и компоненты C2PSA с параллельным пространственным вниманием. Эти улучшения обеспечивают повышенную точность и скорость вывода. Однако, несмотря на оптимизации, полноценные версии YOLOv11 остаются вычислительно затратными для развертывания на мобильных устройствах, встроенных системах и граничных вычислительных

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 открытые наборы.

A. Открытый датасет car-detection

Набор [5] изображений охватывает широкий спектр ракурсов, дистанций, условий освещённости и категорий транспортных средств — от легковых автомобилей до грузовиков и автобусов, — что обеспечивает высокий уровень разнообразия визуальных характеристик и способствует повышению обобщающей способности моделей компьютерного зрения. На рисунках 1, 2 отображены некоторые нетривиальные примеры данных:

Рис. 1. Примеры кадров из открытого датасета

Рис. 2. Пример кадра из открытого датасета

B. Собственный датасет

Помимо открытых датасетов, в данной работе был собран и размечен собственный датасет, содержащий 1072 высококачественных изображений автомобилей, снятых в различных условиях дорог. Датасет включает изображения, снятые с различных типов камер: стандартные видеокамеры, камеры видеонаблюдения.

Разметка была выполнена вручную с помощью ресурса roboflow с последующей проверкой качества. Для каждого автомобиля были определены ограничивающие прямоугольники. Собственный датасет собран на данных, максимально приближенных к реальным условиям использования. Данные выложены в открытый доступ [6]

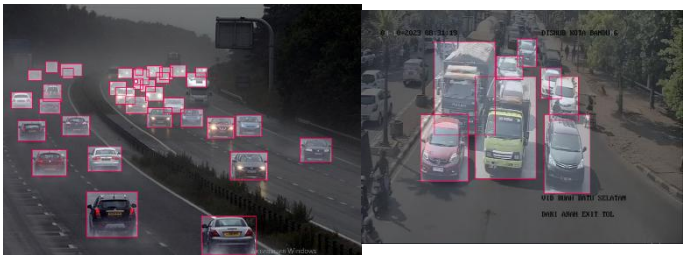


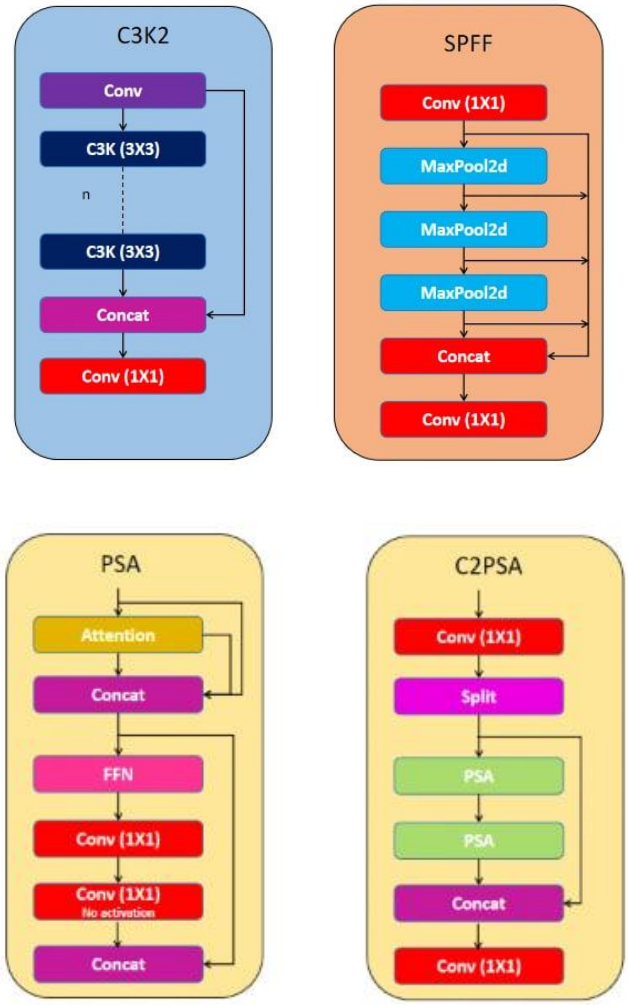
Рис. 3. Примеры кадров из собственного датасета

III. АРХИТЕКТУРА YOLO

YOLOv11 является последним поколением в семействе детекторов, представляющим значительный прогресс в архитектурном дизайне. Модель базируется на улучшенной версии архитектуры Darknet с введением новых модулей. Ключевым нововведением является блок C3k2 (Cross Stage Partial с размером ядра 2), который обеспечивает более эффективное извлечение признаков с использованием разреженных сверток. Блок SPPF (Spatial Pyramid Pooling - Fast) в шеи сети позволяет агрегировать пространственные признаки на нескольких масштабах без значительного увеличения вычислительной нагрузки.

Важным нововведением является компонент C2PSA (Convolutional block with Parallel Spatial Attention), который интегрирует механизмы пространственного внимания прямо в процесс извлечения признаков. Это позволяет сети сосредоточиться на наиболее релевантных регионах изображения и учитывать сложные пространственные отношения между объектами.

YOLOv11 предоставляет пять вариантов архитектуры



с различными глубинами и шириной сети, позволяя выбрать оптимальный баланс между точностью и скоростью вывода. В данной работе детальнее будут рассмотрены две архитектуры, представленные в таблице I:

Рис. 4. Отличительные блоки архитектуры YOLO11v

ТАБЛИЦА I. Варианты архитектуры YOLOv11

Модель	Скорость CPU ONNX (мс)	Скорость T4 (мс)	Количество параметров (M)
YOLO11n	56.1 ± 0.8	1.5 ± 0.0	2.6
YOLO11l	238.6 ± 1.4	6.2 ± 0.1	25.3

IV. KNOWLEDGE DISTILLATION: ТЕОРИЯ И МЕТОДЫ

Knowledge Distillation является методом передачи знаний из одной нейронной сети в другую [7] Идея состоит в том, чтобы обучить более компактную модель-студента воспроизводить поведение более крупной и точной модели-учителя. Это позволяет достичь

значительного сокращения размера и вычислительной сложности модели при сохранении близкой к исходной точности.

A. *Стандартный метод Knowledge Distillation*

Классический подход Knowledge Distillation, предложенный Хинтоном и соавторами, основан на минимизации расхождения между распределениями вероятностей на выходе моделей учителя и студента. Процесс обучения разделяется на два этапа:

- Этап 1: Обучение модели-учителя. На этом этапе полноценная модель обучается на размеченном датасете с использованием стандартной функции потерь (кросс-энтропия для задач классификации, комбинированная потеря для детекции объектов). Обученная модель-учитель сохраняется для последующего использования при обучении студента.
- Этап 2: Дистилляция в модель-студента. На этом этапе компактная модель-студент обучается двум целям одновременно: минимизировать потери на исходной задаче (hard targets) с использованием разметки датасета; минимизировать расхождение между собственными предсказаниями и полученными от модели-учителя.

Предсказания от учителя генерируются путем применения функции softmax с повышенной температурой T к логитам модели-учителя. Повышенная температура делает распределение вероятностей более "мягким", сглаживая различия между классами.

Классическая функция потерь Knowledge Distillation для *классификации* определяется как:

$$\mathcal{L} = \alpha \mathcal{L}_{CE}(y, p_s) + (1-\alpha) T^2 \cdot \mathcal{L}_{KL}(p_t/T, p_s/T), \quad (1)$$

где p_t и p_s — softmax-вероятности учителя и студента, T — это некоторый параметр, условно называемый «температурой», потому что он управляет «резкостью» (концентрацией) распределения вероятностей после softmax, y — истинные метки, α — балансирующий коэффициент (часто 0.1–0.9), $\mathcal{L}_{CE}(y, p_s)$ — кросс-энтропия, $\mathcal{L}_{KL}(p_t/T, p_s/T)$ – KL-дивергенция

Использование температуры T > 1 критически важно, так как она позволяет модели-учителю передать информацию о неправильных ответах и их относительной правдоподобности, которые могут содержать полезную информацию для улучшения студента.

B. *Channel-Wise Distillation*

Channel-Wise Knowledge Distillation (CKD) — это инновационный подход к передаче знаний, предложенный в работе "Channel-wise Knowledge Distillation for Dense Prediction".[8] В отличие от традиционных методов дистилляции знаний, которые выравнивают активационные карты в пространственной области (по пикселям), CKD нормализует активационные карты по каналам, преобразуя их в вероятностные распределения. Ключевая особенность этого подхода заключается в том, что метод уделяет особое внимание наиболее значимым регионам каждого канала, которые содержат сценические признаки или объекты переднего плана. Это позволяет студенческой сети лучше сосредоточиться на критически важных активациях, вместо того чтобы равномерно выравнивать все пиксели и каналы.

Основная формула CKD основана на минимизации асимметричной дивергенции Кульбака-Лейблера (KL divergence) [9]. Сначала активационные карты нормализуются для каждого канала с помощью функции softmax:

$$P_c = \text{softmax}\left(\frac{y_c}{T}\right) = \frac{\exp(y_{c,i}/T)}{\sum_{i=1}^{W \cdot H} \exp(y_{c,i}/T)}, \quad (2)$$

где c — индекс канала, i — индекс пространственной позиции, $W \times H$ — размерность карты активаций, а T — параметр температуры. Затем функция потерь вычисляется как KL дивергенция между нормализованными распределениями учителя и студента:

$$L_{\text{KL}} = \frac{1}{C} \sum_{c=1}^{C} \sum_{i=1}^{W \cdot H} P_T^{c,i} \log \frac{P_T^{c,i}}{P_S^{c,i}}, \quad (3)$$

где P_T и P_S — нормализованные вероятностные карты учителя и студента соответственно, а C — количество каналов.

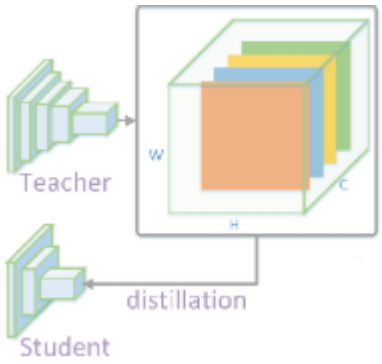


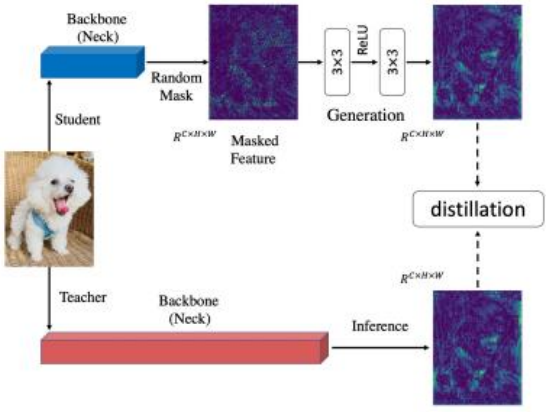
Рис. 5. Ахитектура метода Channel-Wise Distillation

Асимметричная природа KL дивергенции играет критическую роль в эффективности метода. Когда активация учителя $P_T^{c,i}$ велика, студент должен воспроизвести аналогичную величину; однако, если $P_T^{c,i}$ мала, KL дивергенция уделяет меньше внимания минимизации активации студента. Это свойство позволяет сети естественным образом различать регионы переднего плана и фона, что особенно полезно для задач плотного предсказания, таких как семантическая сегментация и детектирование объектов. Экспериментальные результаты показали значительные улучшения производительности: например, метод улучшил детектор RetinaNet с ResNet50 на 3.4 mAP на датасете COCO и PSPNet с ResNet18 на 5.81 mIoU на датасете Cityscapes.

C. *Masked Generative Distillation*

Masked Generative Distillation (MGD) — это метод дистилляции знаний на основе признаков (feature-based distillation), который заставляет студенческую сеть генерировать полную карту признаков учителя из маскированной (частично скрытой) карты признаков

студента, вместо прямого копирования признаков учителя. Ключевое отличие MGD от других методов дистилляции заключается в том, что вместо того, чтобы заставлять студента максимально похоже воспроизводить признаки учителя, метод маскирует случайные пиксели в признаках студента и требует восстановить полные признаки учителя через простой блок (два 3×3 конволюционных слоя с ReLU). Авторы



основывают свой подход на гипотезе, что признаки глубоких слоёв нейронной сети содержат информацию о соседних пикселькаx, поэтому, используя только частичную информацию, можно восстановить всю карту признаков, одновременно улучшая репрезентационную способность сети.

Рис. 6. Ахитектура метода Masked Generative Distillation

Формально, метод использует **несколько ключевых** формул. Во-первых, создаётся случайная маска для каждого слоя l:

$$M_{i,j}^{l} = \begin{cases} 0, & \text{если } R_{i,j}^{l} < \lambda \\ 1, & \text{иначе} \end{cases}, \quad (4)$$

где $R_{i,j}^{l}$ — случайное число в диапазоне (0, 1), а λ — гиперпараметр, определяющий отношение маскирования. Затем студент должен восстановить полные признаки учителя, используя маскированные признаки через генеративный блок $\mathcal{G}$:

$$\mathcal{G}(f_{align}(S^{l}) \cdot M^{l}) \longrightarrow T^{l}, \quad (5)$$

где S^{l} — признаки студента, T^{l} — признаки учителя, а f_{align} — адаптационный слой для выравнивания размерностей. Генеративный блок имеет простую структуру: $\mathcal{G}(F) = W_{l2}(ReLU(W_{l1}(F)))$, где W_{l1} и W_{l2} — свёрточные слои размером 3×3.

Финальная **функция потерь дистилляции** для MGD записывается как:

$$L_{dis}(S,T) = \sum_{l=1}^{L} \sum_{k=1}^{C} \sum_{i=1}^{H} \sum_{j=1}^{W} (T_{k,i,j}^{l} - \mathcal{G}(f_{align}(S_{k,i,j}^{l}) \cdot M_{i,j}^{l}))^2, \quad (6)$$

где L — количество слоёв для дистилляции, C, H, W — размеры карты признаков (количество каналов, высота и ширина). Общая функция потерь при обучении составляет $L_{all} = L_{original} + \alpha \cdot L_{dis}$, где α — гиперпараметр балансировки.

V. Сравнительный анализ методов

Для проведения экспериментов по дистилляции знаний были выбраны две модели из семейства YOLOv11:

Модель-учитель: YOLO11L. Модель-учитель была обучена на комбинированном датасете, Предварительная подготовка модели-учителя проводилась с использованием стандартной функции потерь для задач детекции объектов.

Модель-студент: YOLO11n. Выбор обоснован целью демонстрации возможности существенного сжатия модели при сохранении приемлемых метрик точности детекции. Модель-студент инициализировалась случайными весами перед процессом дистилляци.

Для оценки качества моделей были выбраны две метрики:

Метрика **mAP (mean Average Precision)** является стандартным показателем качества в задачах детекции объектов [10], отражающим среднее значение **Average Precision (AP)** по всем классам. Для каждого класса c метрика **AP** определяется как интеграл от функции **precision** $P(r)$ по **recall** r:

$$AP_c = \int_0^1 P_c(r)\, dr, \quad (7)$$

где $P_c(r)$ описывает зависимость точности от полноты для данного класса после ранжирования предсказаний по уверенности модели. Среднее значение по всем классам даёт **mAP**:

$$mAP = \frac{1}{N} \sum_{c=1}^{N} A\,P_c, \quad (8)$$

где N — количество классов.

В контексте моделей типа YOLO, **mAP50** вычисляется при фиксированном пороге **IoU = 0.50**, где **IoU (Intersection over Union)** определяется как

$$IoU = \frac{|B_{pred} \cap B_{gt}|}{|B_{pred} \cup B_{gt}|}, \quad (9)$$

где B_{pred} и B_{gt} — предсказанный и истинный ограничивающие прямоугольники соответственно. Таким образом, **mAP50** измеряет среднюю точность при условии, что предсказания пересекаются с истинными регионами не менее чем на 50%, и широко применяется для оценки моделей в условиях, где допустима умеренная локализационная погрешность.

Метрика **mAP50–95** представляет собой более строгий и комплексный показатель, усреднённый по 10 порогам $IoU \in \{0.50, 0.55, 0.60, \ldots, 0.95\}$:

$$mAP_{50-95} = \frac{1}{10} \sum_{t=0.50}^{0.95} mA\,P_t, \quad (10)$$

что позволяет оценить способность модели сохранять точность при разных уровнях строгости локализации.

Для проведения экспериментов по сравнению различных методов дистилляции знаний была реализована единая экспериментальная среда на базе

фреймворка PyTorch и библиотеки Ultralytics. Все эксперименты проводились на одинаковом вычислительном оборудовании (GPU NVIDIA RTX 4090 super) с фиксированными гиперпараметрами обучения. Процесс экспериментирования включал следующие этапы:

Модель YOLO11L была обучена на комбинированном датасете, включающем открытые датасеты и собственный размеченный датасет, в течение 100 эпох. Обученная модель-учитель продемонстрировала следующие показатели на тестовом наборе: mAP50 = 0.77129 и mAP50-95 = 0.53994.

Также была обучена модель YOLO11n с использованием аналогичной конфигурации и датасетов, без методов дистилляции. Это позволило установить базовый уровень производительности и оценить выгоду от применения методов Knowledge Distillation.Функция потерь на валидационной выборке после 80 эпох перестала снижаться, что обучение уперлось в плато. Базовая модель-студент показала результаты: mAP50 = 0.73104 и mAP50-95 = 0.50246.

Модель-студент была переобучена с применением двух различных методов дистилляции. Для метода Channel-Wise Knowledge Distillation (CKD) функция потерь была определена как комбинация стандартной детекционной потери и потери CKD. Для метода Masked Generative Distillation (MGD) модель обучалась с применением генеративного блока для восстановления полных карт признаков из маскированных входов. Обучение проводилось в течение 200 эпох с динамическим уменьшением скорости обучения. Метод CKD достигнул следующих показателей: mAP50 = 0.74830 и mAP50-95 = 0.52379. Метод MGD показал результаты: mAP50 = 0.74367 и mAP50-95 = 0.52042.

ТАБЛИЦА II. Результаты экспериментов

Модель	**mAP50**	**mAP50–95**
YOLO11l	0.77129	0.53994
YOLO11n	0.73104	0.50246
YOLO11n + CKD	0.74830	0.52379
YOLO11n + MGD	0.74367	0.52042

VI. Заключение

В настоящей работе выполнена экспериментальная оценка применимости методов Knowledge Distillation к семейству моделей YOLOv11 в задаче детекции автомобилей. Полученные результаты показывают, что при достижении базовой моделью-студентом удовлетворительного уровня качества использование дистилляции может рассматриваться как эффективный способ дальнейшего повышения метрик без перехода к более вычислительно затратным архитектурам. В рамках исследования проанализированы два подхода — Channel-Wise Knowledge Distillation (CKD) и Masked Generative Distillation (MGD)

Установлено, что оба рассматриваемых метода обеспечивают улучшение показателей детекции по сравнению с недистиллированной моделью-студентом при сохранении ее компактности. Дополнительно показано, что перенос знаний от более крупной модели-учителя позволяет получать выигрыши в качестве без увеличения числа параметров у модели-студента. В проведенных экспериментах достигнуто сжатие модели на 89.7%: количество параметров уменьшено с 25.3 млн (YOLO11L) до 2.6 млн (YOLO11n). Снижение размерности модели порядка 90% приводит к уменьшению вычислительной сложности и, как следствие, потенциально снижает энергопотребление при инференсе.

Практическая значимость результатов заключается в подтверждении целесообразности применения Knowledge Distillation для построения ресурсно-эффективных детекторов объектов, обеспечивающих повышенные метрики качества при ограниченных вычислительных ресурсах. Это делает дистиллированные модели перспективными для использования в системах с жесткими ограничениями по памяти и энергопотреблению, включая мобильные и встраиваемые платформы.

Литература

[1] Епифанов Владислав Александрович, Темкин Игорь Олегович, Красноярский Сергей Евгеньевич ЭФФЕКТИВНЫЙ АЛГОРИТМ ИДЕНТИФИКАЦИИ ТРАНСПОРТНЫХ СРЕДСТВ В СИСТЕМАХ ВИДЕОНАБЛЮДЕНИЯ // ГИАБ. 2023. №6. URL: https://cyberleninka.ru/article/n/effektivnyy-algoritm-identifikatsii-transportnyh-sredstv-v-sistemah-videonablyudeniya.

[2] Епифанов Владислав Александрович Применение методов искусственного интеллекта и машинного зрения в задаче детектирования объектов движения для дальнейшего определения динамических характеристик транспортных потоков // Евразийский Союз Ученых. 2019. №4-3 (61). URL: https://cyberleninka.ru/article/n/primenenie-metodov-iskusstvennogo-intellekta-i-mashinnogo-zreniya-v-zadache-detektirovaniya-obektov-dvizheniya-dlya-dalneyshego.

[3] Алексеев, Л. Е. Распознавание самокатов в реальном времени с помощью YOLO: сравнительный анализ YOLOv10, YOLOv11 / Л. Е. Алексеев // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 4-7. – EDN TVNDGY.

[4] Khanam R., Hussain M. YOLOv11: An Overview of the Key Architectural Enhancements [Электронный ресурс] // arXiv, 23.10.2024. № 2410.17725. URL: https://arxiv.org/abs/2410.17725

[5] car-detection Dataset [Электронный ресурс] / Drink Tracker // Roboflow Universe. – 2025. – URL: https://universe.roboflow.com/drink-tracker/car-detection-pn2gk

[6] CCTV_cars_dataset [Электронный ресурс] / ArtyomTsykanov // Hugging Face. – 2025. – URL: https://huggingface.co/datasets/ArtyomTsykanov/CCTV_cars_dataset

[7] Hinton G., Vinyals O., Dean J. Distilling the Knowledge in a Neural Network [Электронный ресурс] // arXiv. – 2015. – № 1503.02531. – URL: https://arxiv.org/abs/1503.02531

[8] Shu C., Liu Y., Gao J., Yan Z., Shen Ch. Channel-wise Knowledge Distillation for Dense Prediction // Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). – 2021. – P. 5291–5300.

[9] Лемешко Борис Юрьевич, Лемешко Станислав Борисович О свойствах и проблемах применения критериев согласия, опирающихся на использование оценок энтропии и дивергенции Кульбака – Лейблера // Системы анализа и обработки данных. 2025. №2 (98). URL: https://cyberleninka.ru/article/n/o-svoystvah-i-problemah-primeneniya-kriteriev-soglasiya-opirayuschihsya-na-ispolzovanie-otsenok-entropii-i-divergentsii-kulbaka.

[10] Базалеев, Ф. Е. Исследование возможности детектирования дорожных знаков на основе нейросетевой модели YOLO / Ф. Е. Базалеев, Е. И. Пиховская // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 30-34. – EDN CDLGYL..

ГРНТИ 28.23.15, 28.23.37, 50.41.29
ASJC 1702, 1706
OECD 1.02 EP

Нейросетевые методы для детектирования элементов пользовательского интерфейса

Е. С. Четвертков
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2105365@edu.misis.ru

***Аннотация* — в статье рассматриваются нейросетевые методы детектирования элементов пользовательского интерфейса на основе современных одностадийных детекторов семейства YOLO. В работе проведено сравнение моделей YOLOv11, YOLOv12 и YOLOv13 в контексте распознавания UI-элементов, таких как кнопки, поля ввода, текстовые блоки и изображения. Для обучения моделей был сгенерирован синтетический датасет веб-интерфейсов с автоматической разметкой на основе DOM-дерева, а также использован небольшой набор реальных скриншотов для дообучения и валидации. Проведен анализ влияния архитектурных решений, включая механизмы внимания и гиперграфовые корреляции, на точность и скорость инференса. Экспериментальные результаты демонстрируют, что модель YOLOv13 обеспечивает наилучший баланс между качеством детекции и вычислительной эффективностью, что делает ее перспективной для практического применения в задачах автоматизированного анализа пользовательских интерфейсов.**

Ключевые слова — компьютерное зрение, распознавание элементов интерфейса, глубокое обучение, YOLOv11, YOLOv12, YOLOv13, синтетические данные.

I. Введение

В последние годы задача автоматического распознавания элементов пользовательского интерфейса вышла далеко за рамки классического компьютерного зрения, став фундаментальным компонентом для целого ряда прикладных направлений в разработке программного обеспечения и искусственного интеллекта. Во-первых, детектирование виджетов играет ключевую роль в системах генерации кода, где методы глубокого обучения транслируют визуальные макеты в исполняемый код [1, 2]. Другим таким применением является автоматизированное тестирование, в котором детекция виджетов повышает эффективность стратегий покрытия тестами [3]. Смежные задачи, такие как распознавание формул [4] и таблиц [5] также являются востребованными и получают активное развитие в последнее время.

Однако, наиболее значимый импульс развитию этой области придал недавний взрывной рост агентных систем на базе больших языковых моделей. Современные мультимодальные агенты стремятся взаимодействовать с компьютером так же, как это делает человек – через визуальный канал восприятия. Для этого модели должны не просто “видеть” экран, но и точно интерпретировать его функциональные зоны. Проекты, подобные OmniParser [6], демонстрируют, что методы предварительного парсинга визуальных элементов позволяют значительно повысить эффективность взаимодействия модели с интерфейсом. Все это позволяет создавать цифровых помощников, способных выполнять сложные пользовательские сценарии.

Специфика анализа интерфейсов существенно отличается от классического распознавания объектов реального мира. Изображения интерфейсов характеризуются высокой плотностью элементов, их визуальной схожестью и сложной иерархией. Виджеты часто накладываются друг на друга, имеют малые размеры или располагаются вплотную, что требует от алгоритмов детекции высокой точности локализации границ, недостижимой для многих стандартных подходов.

Отдельной сложностью при подготовке нейросетевых решений является создание качественной разметки реальных данных. Большое количество элементов на одном изображении, а также огромная вариативность визуальных реализаций – от различий в дизайн-системах операционных систем до бесконечного множества стилей веб-приложений – делает процесс ручной аннотации крайне трудоемким и подверженным ошибкам. Автоматическая аннотация также затруднена – часто один и тот же визуальный элемент может быть представлен абсолютно разным исходным кодом, а в современных веб-приложениях кроме того необходимо учитывать различия в z-индексе, фильтры и прозрачность, чтобы не загрязнять датасет визуально неразличимыми или невидимыми элементами. Эта проблема была успешно преодолена благодаря внедрению методов генерации синтетических данных. Использование программных движков для рендеринга интерфейсов по заданным шаблонам позволяет автоматически получать неограниченные объемы данных с идеально точными координатами и метаданными, что обеспечивает необходимую надежность при обучении глубоких архитектур [7]. Также, это позволяет создать датасет с любым требуемым набором классов под конкретную задачу без необходимости повторной разметки.

Для решения этих проблем в работе рассматривается эволюция современных одностадийных детекторов семейства YOLO. Модель YOLOv11 [8] отличается

усовершенствованной структурой блоков извлечения признаков, обеспечивая оптимальный баланс между производительностью и точностью, что делает ее эффективной для широкого спектра вычислительных платформ. В то же время архитектура YOLOv12 [9] интегрирует механизмы пространственного внимания и оптимизированную основную сеть R-ELAN, что позволяет лучше улавливать контекстные зависимости сцены без ущерба для скорости реального времени. Дальнейшим развитием подхода стала модель YOLOv13 [10], которая преодолевает ограничения локальной агрегации информации благодаря внедрению гиперграфового механизма HyperACE. Этот метод позволяет учитывать глобальные многосторонние корреляции между объектами, достигая высоких результатов при сниженной вычислительной сложности.

II. НАБОРЫ ДАННЫХ

Для обучения и валидации модели обнаружения объектов был сгенерирован синтетический датасет, состоящий из 5000 аннотированных изображений веб-интерфейсов.

Генерация данных осуществлялась с использованием специально разработанного алгоритма, который имитирует процесс современной веб-разработки. Подход базируется на методологии Atomic Design, где сложные интерфейсы собираются из иерархических компонентов:

- атомы – базовые неделимые элементы (элементы управления, текстовые блоки, изображения);
- молекулы – группы атомов, формирующие функциональные блоки (карточки товаров, формы регистрации, панели навигации, таблицы);
- организмы – финальные страницы, состоящие из молекул.

Для обеспечения визуального разнообразия применялся комплексный подход к рандомизации параметров. Каждая страница получает уникальную визуальную тему, определяющую цветовую палитру, типографику и геометрию элементов (например, радиусы скругления). Наполнение интерфейсов (заголовки, тексты, пользовательские данные) создаётся при помощи генерации случайных слов и предложений на двух языках (английском и русском), что позволяет имитировать реалистичный контент. Техническая вариативность достигается за счет симуляции просмотра на различных устройствах: размеры вьюпорта варьируются в диапазоне от 375px до 1920px с добавлением случайного шума, покрывая как мобильные, так и десктопные сценарии.

Разметка данных производилась автоматически в процессе рендеринга страниц через headless-браузер (playwright) – истинные границы объектов извлекались непосредственно из DOM-дерева, с использованием Intersection Observer для расчета координат только визуально доступных частей элементов. Примеры полученных изображений представлены на рисунке 1.

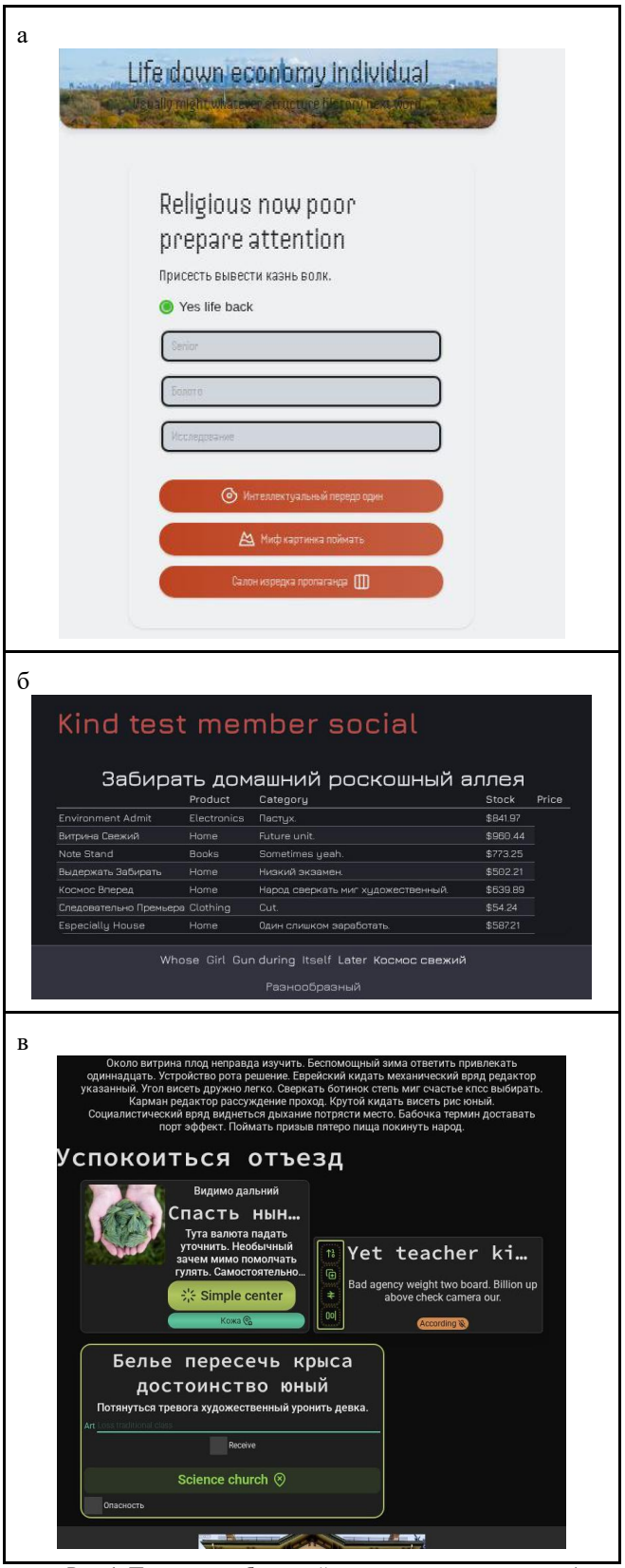


Рис. 1. Примеры изображений в синтетическом датасете: а) структурированная форма; б) таблица; в) карточки с кнопками и текстом.

Датасет включает 6 классов UI элементов: button (кнопки различных стилей), input (поля ввода, выпадающие списки), checkbox (элементы выбора и

переключатели), image (изображения, аватары и баннеры), header (визуальные заголовки), text (текстовые параграфы, лейблы, описания). Распределение классов представлено на рисунке 2.

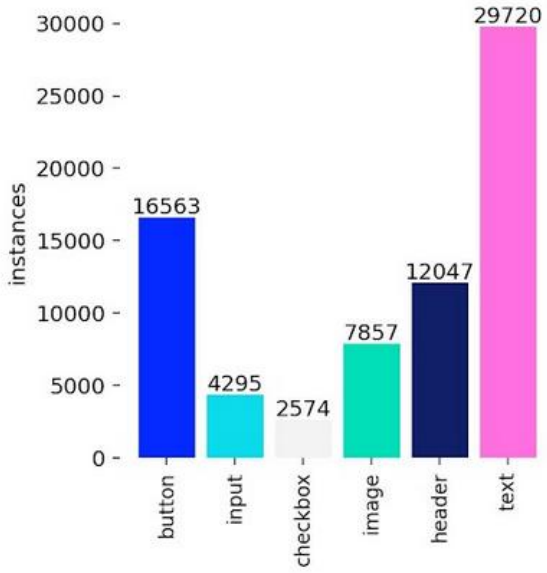


Рис. 2. Распределение классов в синтетическом датасете.

Датасет был разбит на обучающую и валидационную выборку в отношении 90 к 10.

Кроме того, для валидации и дообучения базовых моделей применялся вручную отобранный и размеченный датасет скриншотов реальных сайтов. Датасет содержит 230 размеченных изображений, из них 185 отведены под дообучение, а 45 под валидацию. Распределение классов в этом датасете представлено на рисунке 3.

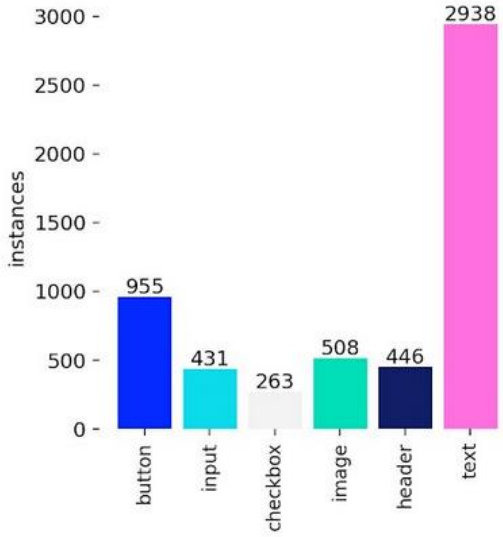


Рис. 3. Распределение классов в реальном датасете.

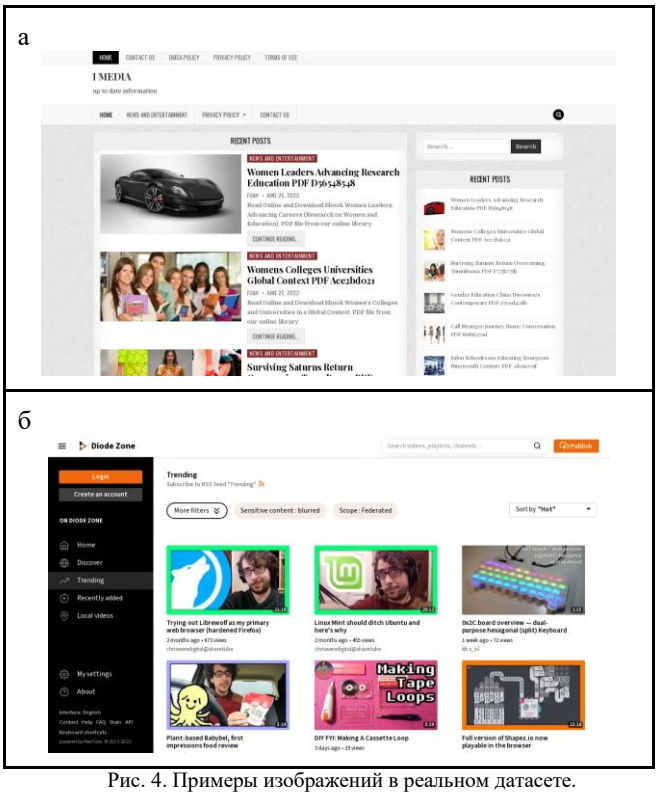


Рис. 4. Примеры изображений в реальном датасете.

Оба датасета доступны публично на HuggingFace [11, 12].

III. Нейросетевые архитектуры

В данной работе рассматриваются три модели из серии YOLO: YOLOv11, YOLOv12 и YOLOv13, каждая из которых предлагает уникальные подходы к извлечению признаков из изображений.

A. YOLOv11

Архитектура YOLOv11 представляет собой развитие идей, заложенных в предыдущих версиях, с упором на повышение эффективности извлечения признаков при сохранении высокой скорости инференса. Ключевым нововведением стало внедрение блока C3k2 (Cross Stage Partial with kernel size 2), который заменил используемый ранее блок C2f. C3k2 обеспечивает более эффективную агрегацию признаков благодаря оптимизированному механизму остаточных связей и использованию вместо одной крупной свертки двух сверток меньшего размера, что положительно сказывается на вычислительной эффективности.

Важным дополнением в основной части модели стало включение модуля C2PSA (Convolutional block with Parallel Spatial Attention) после блока пространственного пирамидального пулинга (SPPF). C2PSA реализует механизм пространственного внимания, позволяя модели фокусироваться на наиболее информативных областях изображения и игнорировать фоновый шум. Это значительно улучшает способность детектора распознавать объекты в условиях частичного перекрытия и сложного фона. В целом, YOLOv11 демонстрирует улучшенный баланс между точностью и количеством параметров по сравнению с предшественниками [13].

B. YOLOv12

Отличительной чертой YOLOv12 является использование механизма внимания, что позволяет преодолеть ограничения классических сверточных сетей в моделировании глобальных зависимостей. В основе архитектуры лежит новый backbone – Residual Efficient Layer Aggregation Network (R-ELAN), который оптимизирует градиентные потоки и улучшает слияние признаков за счет глубоких остаточных связей. Для сохранения пространственного контекста при снижении вычислительной нагрузки в модели применяются разделяемые свертки размером 7x7 [14, 15].

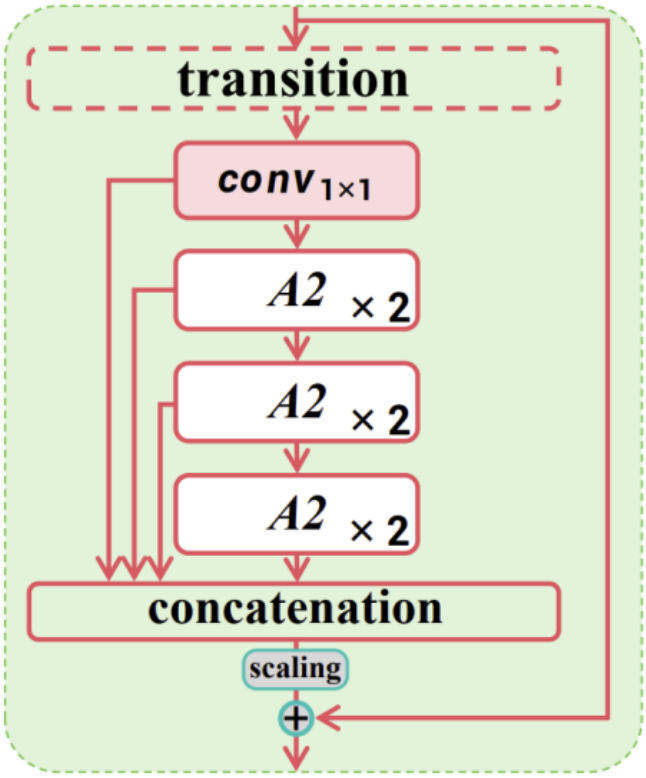


Рис. 6. Схема устройства R-ELAN.

В YOLOv12 используется механизм внимания Area Attention с ускорением через FlashAttention. В отличие от стандартного Self-Attention, который вычислительно дорог на высоких разрешениях, Area Attention сегментирует карты признаков, позволяя эффективно моделировать зависимости в широком рецептивном поле без чрезмерных затрат памяти. Это позволяет модели лучше улавливать контекстуальную информацию и мелкие детали, что критично для задач детекции малых объектов.

C. YOLOv13

В отличие от предыдущих версий, вместо попарного моделирования визуальных признаков YOLOv13 способна анализировать высокоуровневые корреляции типа “многие-ко-многим”. Вместо Area Attention и FlashAttention используемых в YOLOv12, YOLOv13 внедряет парадигму вычислений на основе гиперграфов что позволяет достичь глобального восприятия сцены.

Ключевым компонентом архитектуры является механизм адаптивного усиления корреляций на основе гиперграфов (Hypergraph-based Adaptive Correlation Enhancement, HyperACE). В то время как сверточные сети в YOLO11 ограничены локальными рецептивными полями, а механизмы внимания в YOLOv12 – попарными связями, HyperACE использует адаптивную генерацию гиперребер. Это позволяет модели выявлять скрытые сложные зависимости между множеством пространственных позиций и масштабов одновременно, что значительно повышает точность детекции в перекрытых или визуально перегруженных сценах.

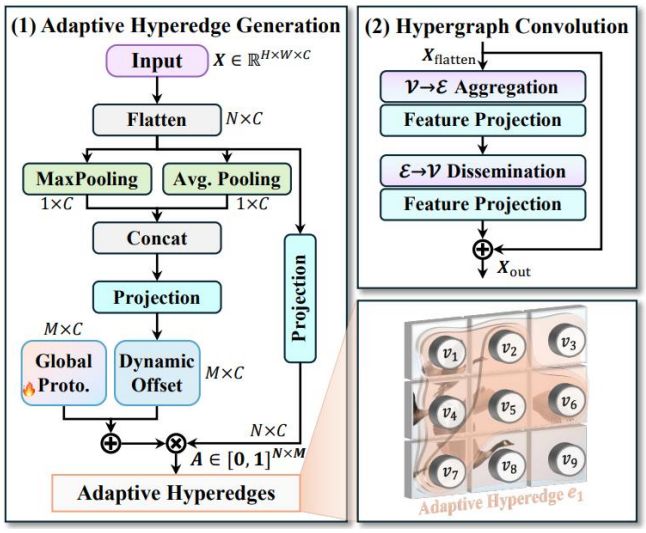


Рис. 7. Механизм создания гиперграфа и определения принадлежности пикселей к гиперребрам.

Для эффективного управления информационными потоками в YOLOv13 реализована парадигма агрегации и распределения по всему конвейеру (Full-Pipeline Aggregation-and-Distribution, FullPAD). В отличие от стандартной последовательной схемы “backbone – neck – head”, FullPAD собирает корреляционно-усиленные признаки из модуля HyperACE и через специализированные туннели распределяет их по всей сети. Это обеспечивает синергию представлений данных на всех этапах обработки и улучшает распространение градиентов при обучении.

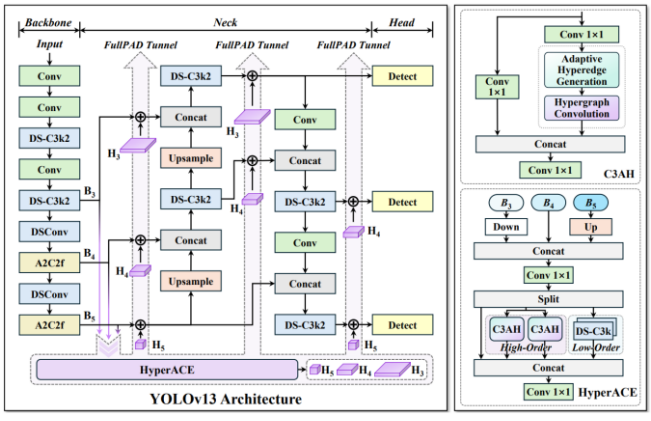


Рис. 8. Архитектура YOLOv13.

Масштабируемость и вычислительная эффективность модели достигнуты за счет внедрения серии облегченных блоков, основанных на глубоко разделяемых свертках (Depthwise Separable Convolutions). В частности, блоки DS-C3k2 и DS-C3k заменяют классические сверточные слои с большими ядрами в основной сети (backbone) и промежуточных слоях (neck). Это позволило сократить количество параметров до 30% и снизить объем вычислений до 28% по сравнению с YOLO11 и YOLOv12 при сохранении и даже росте точности.

IV. Обучение и результаты

В данной работе рассматривались модели размера “small” из семейств YOLOv11, YOLOv12 и YOLOv13, обладающие сопоставимым количеством параметров и хорошим соотношением производительности и точности. Для инициализации моделей использовались веса, полученные на датасете COCO.

Процесс обучения был разделен на два этапа. На первом исходные модели обучались в течение 50 эпох на полностью синтетическом датасете. После чего полученные модели дообучались на смешанном наборе данных, состоящем из 185 реальных и 200 синтетических изображений, добавленных для стабилизации обучения на небольшом датасете. Валидация на этом этапе проводилась исключительно на реальных данных – 45 скриншотах из валидационной выборки.

Для оценки качества работы моделей использовались метрики Precision, Recall и mAP@50-95 [16], а также среднее время обработки одного изображения на видеокарте NVIDIA L4. В замере времени обработки отображено только среднее время работы модели, без учета пред- и пост-обработки.

Для адаптации к специфике предметной области были внесены следующие изменения в стандартный пайплайн обучения YOLO:

1. Размер входного изображения увеличен с 640×640 до 1280×1280 пикселей для улучшения детекции малых объектов, таких как текст и переключатели.
2. Техника Mosaic была отключена, так как нарушение пространственного контекста страницы негативно сказывалось на обучении.

Обучение проводилось с использованием фреймворка Ultralytics на графическом ускорителе NVIDIA A100 (40 GB VRAM). В качестве оптимизатора использовался AdamW с начальным темпом обучения равным 0.001. На этапе дообучения темп был снижен до 0.0003 для предотвращения переобучения на малой выборке. Кроме того, размер батча на этом этапе был снижен с 16 до 8.

Динамики дообучения моделей на реальном датасете представлены на рисунках 9-11.

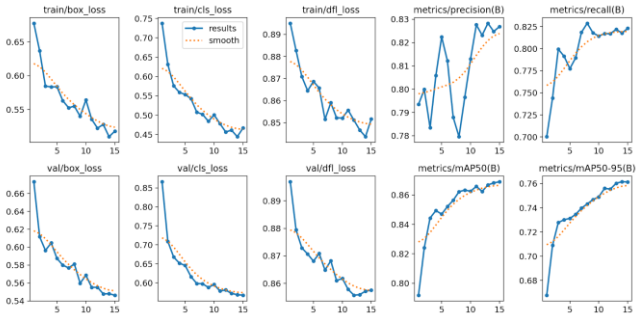


Рис. 9. Кривые обучения YOLOv11s на реальном датасете.

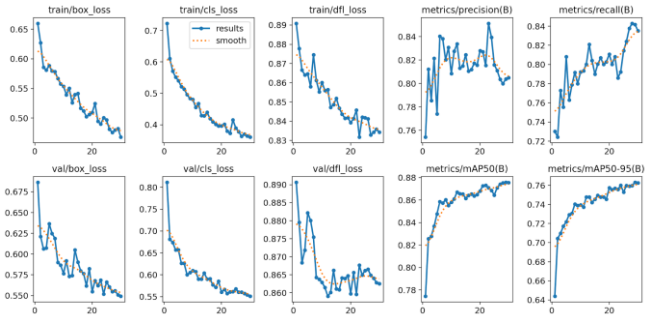


Рис. 10. Кривые обучения YOLOv12s на реальном датасете.

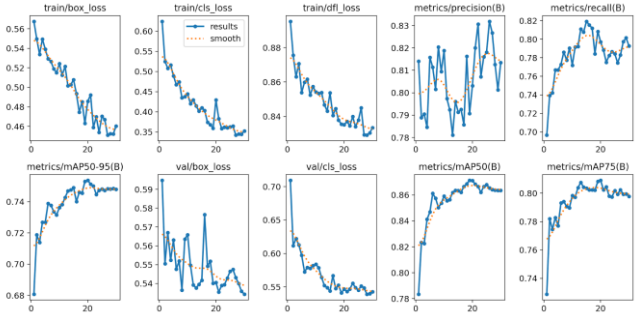


Рис. 11. Кривые обучения YOLOv13s на реальном датасете.

Итоговые результаты полученных моделей на валидационной выборке представлены в таблице 1.

ТАБЛИЦА I. Сравнение результатов

Модель	Precision	Recall	mAP	Время
v11s	0.824	**0.818**	0.771	23.5 мс
v12s	**0.845**	0.801	**0.774**	47.3 мс
v13s	0.829	0.803	0.768	**15.6 мс**

Анализ результатов показывает, что модель YOLOv12 демонстрирует незначительное преимущество по метрикам Precision (0.845) и mAP (0.774). Однако, время инференса данной модели (47.3

мс) является критическим недостатком, превышая показатели YOLOv11 в 2 раза.

Наиболее сбалансированным решением является YOLOv13: благодаря более эффективным архитектурным решениям при сопоставимом с v11 и v12 качестве детекции (mAP 0.768), она обеспечивает наилучшую производительность (15.6 мс), что в 3 раза быстрее YOLOv12. Это делает YOLOv13 оптимальным выбором, снижающим затраты на развертывание и дающим больший простор для прикладного применения.

V. Применение обученной модели

Для демонстрации практической значимости и эффективности разработанного алгоритма распознавания элементов веб-интерфейса было реализовано прикладное программное обеспечение – десктопное приложение, автоматизирующее процесс обратного проектирования пользовательских интерфейсов, позволяя преобразовывать растровые изображения в редактируемые векторные макеты.

Приложение реализовано на базе фреймворка Tauri, который позволяет создавать кроссплатформенное программное обеспечение, объединяя веб-технологии с производительностью системных языков. Пользовательский интерфейс написан на TypeScript с использованием библиотеки React, что обеспечивает гибкость и высокую интерактивность при работе с графическими элементами макета.

Вычислительное ядро программы построено на языке Rust. Непосредственный запуск обученной модели осуществляется через библиотеку ort – Rust-обертку для ONNX Runtime. Такое решение позволяет выполнять все нейросетевые вычисления локально на устройстве пользователя.

Процесс работы с приложением состоит из нескольких этапов:

1. Пользователь загружает изображение интерфейса. Система подготавливает его и подает на вход нейросети. Алгоритм выделяет ключевые UI-компоненты, определяет их координаты и классы. Результат распознавания выводится на экран для предпросмотра (рисунок 12).
2. Если пользователя устраивает результат распознавания, на основе полученных координат и классов приложение реконструирует интерфейс, заменяя оригинальные растровые элементы на стилизованные компоненты и переходит в режим редактирования макета, представленный на рисунке 13.
3. Во встроенном редакторе пользователь может вручную добавить пропущенные элементы, уточнить позиции распознанных, или использовать набросок со скриншота как основу для создания своего интерфейса.
4. Итоговый макет может быть экспортирован в формате изображения.

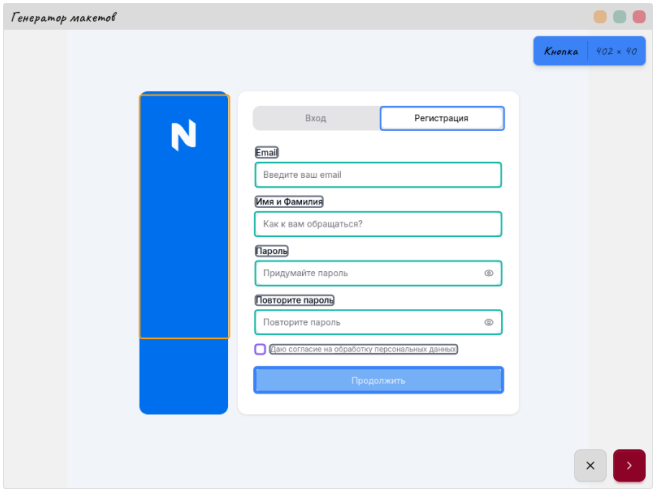


Рис. 12. Предпросмотр найденных на изображении элементов интерфейса.

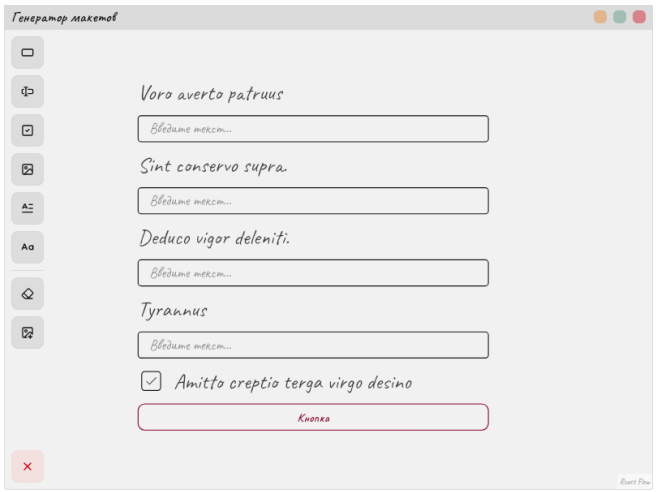


Рис. 13. Экран редактирования макета, созданного на основе распознанных данных.

Разработанное приложение существенно сокращает время, затрачиваемое дизайнерами и аналитиками на перерисовку существующих интерфейсов, позволяя сразу переходить к этапу проектирования улучшений и анализа конкурентных решений.

VI. Заключение

В данной работе были исследованы нейросетевые методы детектирования элементов пользовательского интерфейса на основе современных одностадийных детекторов семейства YOLO. Основное внимание было уделено сравнению архитектур YOLOv11, YOLOv12 и YOLOv13 в задаче распознавания типовых UI-элементов веб-интерфейсов, характеризующихся высокой плотностью объектов, их визуальной схожестью и сложной пространственной структурой.

Для проведения экспериментов был разработан синтетический датасет веб-интерфейсов с автоматической разметкой на основе DOM-дерева, что позволило получить масштабируемый и контролируемый источник обучающих данных. Дополнительно использовался небольшой набор

реальных скриншотов, применённый для дообучения и валидации моделей, что обеспечило более корректную оценку качества работы в приближенных к практическим условиях сценариях.

Экспериментальные результаты показали, что все рассмотренные модели способны эффективно решать задачу детектирования UI-элементов, однако имеют различный баланс между точностью и вычислительной эффективностью. YOLOv12 продемонстрировала наивысшие значения Precision и mAP, что подтверждает эффективность механизмов внимания для анализа сложных визуальных сцен. В то же время высокая вычислительная стоимость данной архитектуры существенно ограничивает её применение в задачах реального времени.

Наиболее практичным решением по совокупности характеристик оказалась модель YOLOv13. Использование гиперграфового механизма HyperACE и схемы FullPAD позволило достичь сопоставимого качества детекции при значительно меньшем времени инференса. Это делает YOLOv13 особенно перспективной для задач автоматизированного анализа пользовательских интерфейсов, мультимодальных агентных систем и инструментов автоматического тестирования, где критичны как точность, так и скорость обработки.

Также на основе обученной модели было разработано приложение для обратного проектирования пользовательских интерфейсов, снижающее затраты на анализ конкурентов, что доказывает практическую значимость данной работы.

Литература

[1] Xu, Y. image2emmet: Automatic code generation from web user interface image / Y. Xu, L. Bo, X. Sun [et al.] // Journal of Software: Evolution and Process. — 2021. — Vol. 33, no. 8. — P. e2369. — DOI: 10.1002/smr.2369.

[2] Chen, W.-Y. Code Generation from a Graphical User Interface via Attention-Based Encoder–Decoder Model / W.-Y. Chen, P. Podstreleny, W.-H. Cheng [et al.] // Multimedia Systems. — 2022. — Vol. 28, no. 1. — P. 121–130. — DOI: 10.1007/s00530-021-00804-7.

[3] White, T. D. Improving Random GUI Testing with Image-Based Widget Detection / T. D. White, G. Fraser, G. J. Brown // Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis (ISSTA 2019). — New York : ACM, 2019. — P. 307–317. — DOI: 10.1145/3293882.3330551.

[4] Катызина, А. А. Распознавание рукописных математических выражений с использованием нейронных сетей / А. А. Катызина, А. Т. Фам // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 63-69. – EDN HPDCOA.

[5] Anand, A. TC-OCR: TableCraft OCR for Efficient Detection & Recognition of Table Structure & Content / A. Anand, R. Jaiswal, P. Bhuyan [et al.] // Proceedings of the 1st International Workshop on Deep Multimodal Learning for Information Retrieval (MM '23). — [S. l.] : ACM, 2023. — P. 11–18. — DOI: 10.1145/3606040.3617444.

[6] Lu, Y. OmniParser for Pure Vision Based GUI Agent / Y. Lu, J. Yang, Y. Shen [et al.]. — 2024. — URL: https://arxiv.org/abs/2408.00203 (дата обращения: 27.12.2025). — Препринт arXiv:2408.00203.

[7] Аскари Хеммат, С. Применение синтетических данных из UnrealEngine для обучения модели сегментации мебели / С. Аскари Хеммат // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 13-16. – EDN FMSJUI.

[8] Khanam, R. YOLOv11: An Overview of the Key Architectural Enhancements / R. Khanam, M. Hussain. — 2024. — URL: https://arxiv.org/abs/2410.17725 (дата обращения: 27.12.2025). — Препринт arXiv:2410.17725.

[9] Tian, Y. YOLOv12: Attention-Centric Real-Time Object Detectors / Y. Tian, Q. Ye, D. Doermann. — 2025. — URL: https://arxiv.org/abs/2502.12524 (дата обращения: 27.12.2025). — Препринт arXiv:2502.12524.

[10] Lei, M. YOLOv13: Real-Time Object Detection with Hypergraph-Enhanced Adaptive Visual Perception / M. Lei, S. Li, Y. Wu [et al.]. — 2025. — URL: https://arxiv.org/abs/2506.17733 (дата обращения: 27.12.2025). — Препринт arXiv:2506.17733.

[11] Chetvertkov, E. S. Synthetic Web UI Dataset. (2025) URL: https://huggingface.co/datasets/egnch/synthetic-ui

[12] Chetvertkov, E. S. Real Web UI Dataset. (2025) URL: https://huggingface.co/datasets/egnch/real-ui

[13] Коротких, И. А. Обнаружение и классификация повреждений костей с использованием нейронных сетей / И. А. Коротких // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 90-96. – EDN SWYUPL.

[14] Ашманова, Е. А. Нейросетевые методы для распознавания дронов и птиц в воздушном пространстве / Е. А. Ашманова, С. В. Старцев // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 17-23. – EDN IMZARK.

[15] Alif, M. A. R. YOLOv12: A Breakdown of the Key Architectural Features / M. A. R. Alif, M. Hussain. — 2025. — URL: https://arxiv.org/abs/2502.14740 (дата обращения: 27.12.2025). — Препринт arXiv:2502.14740.

[16] 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.

ГРНТИ 28.23.15
ASJC 1707
OECD 1.02

Помощник для игры в шахматы на основе технического зрения

П. С. Чикин
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2102734@edu.misis.ru

М. В. Зайцев
кафедра инженерной кибернетики НИТУ «МИСиС»
Москва, Россия
m2106478@edu.misis.ru

Аннотация — в настоящее время активно развиваются интеллектуальные системы для анализа и интерпретации визуального контента в ограниченных, структурированных средах, например таких как шахматная доска. Автоматическое распознавание шахматных фигур и их позиций имеет важное значение для создания цифровых ассистентов и систем анализа партий в реальном времени. Данная задача включает в себя как геометрическое выделение доски на изображении, так и точную детекцию и классификацию фигур. В работе рассматриваются несколько подходов к решению поставленной задачи, основанных на многоэтапных преобразованиях исходного изображения, включая современные методы глубокого обучения: YOLO, RF-DETR, а также архитектуры на основе трансформеров (ViT, ConvNeXt) и мультимодальных моделей (CLIP). Проведено сравнение эффективности различных подходов на реальных наборах данных. Помимо этого, был составлен большой датасет шахматных досок с размеченными фигурами. Результаты демонстрируют высокую точность и практическую применимость предложенного решения.

Ключевые слова — детекция шахматной доски, классификация шахматных фигур, компьютерное зрение, yolo, rf-DETR, vit, convnext, clip.

I. Введение

Искусственный интеллект и компьютерное зрение находят всё более широкое применение. Например, в задачах определения дефектов дорожного покрытия с видеопотока [1], для распознавания стрелочных переводов [2] и задачах распознавания рукописных математических выражений [3]. Особый интерес представляют задачи, связанные с анализом структурированных сред, где геометрия сцены известна. Например, шахматная доска: строго определённая 8×8 сетка, фиксированное множество классов объектов, а также чёткие правила размещения и перемещения. Несмотря на кажущуюся простоту, автоматическое распознавание шахматной позиции по одному или последовательности изображений остаётся нетривиальной задачей из-за разнообразия условий съёмки: изменчивое освещение, перспективные искажения, различия в дизайне фигур, а также частичные перекрытия объектов и отражения на поверхности доски [4].

Задача детектирования и классификации шахматных фигур имеет практическую значимость в ряде прикладных областей: создание ассистентов для анализа партий в реальном времени, автоматизация трансляций турниров, а также образовательные приложения. При этом ключевым требованием к таким системам является не только высокая точность распознавания, но и устойчивость к неидеальным условиям.

В последние годы методы глубокого обучения, особенно модели детекции объектов, такие как YOLO, DETR и его модификации (включая RF-DETR), а также архитектуры на основе трансформеров (ViT, ConvNeXt) и мультимодальные подходы (CLIP), показали высокую эффективность в задачах локализации и классификации объектов в сложных сценах.

В данной работе рассматриваются и сравниваются несколько подходов к решению задачи детектирования шахматной доски и распознавания фигур, включая:

- двухэтапную стратегию с геометрическим выравниванием доски;
- прямую детекцию фигур с использованием YOLO и RF-DETR;
- гибридные методы с применением классификаторов на основе ViT, ConvNeXt и zero-shot подходов с использованием CLIP.

Для оценки предложенных решений был создан размеченный датасет, включающий несколько стилей досок и фигур, ракурсы и условия освещения. Эксперименты показывают, что комбинированный подход, интегрирующий геометрические приоры и современные архитектуры глубокого обучения, обеспечивает наилучший баланс между точностью, скоростью работы и устойчивостью к внешним возмущениям.

II. Набор данных

Для обучения и оценки качества алгоритмов технического зрения использовался собственный набор данных изображений шахматной доски с расставленными фигурами из разных наборов. Разметка данных выполнена с использованием платформы CVAT.ai (https://www.cvat.ai/), что обеспечило единый формат аннотаций и возможность получать разметку во многих форматах данных, таких как COCO или YOLO. Набор данных содержит два типа разметки, ориентированных на решение различных подзадач. Пример размеченного изображения на рисунке 1.

Рис. 1 Пример разметки CVAT

Первый тип разметки предназначен для локализации шахматной доски на изображении и представлен полигональной аннотацией в виде четырёхугольника, соответствующего внешнему контуру доски. Для данной задачи используется единый класс chess_board_polygon (522 объекта), что позволяет выделять игровую область и выполнять последующую сегментацию шахматной доски.

Второй тип разметки используется для задачи детектирования и классификации шахматных фигур и представлен прямоугольными ограничивающими рамками. Каждая фигура аннотирована отдельным классом с учётом её типа и цвета. В датасете представлены следующие классы:

- chess_bishop_black (615 объектов);
- chess_bishop_white (557 объектов);
- chess_king_black (413 объектов);
- chess_king_white (405 объектов);
- chess_knight_black (413 объектов);
- chess_knight_white (573 объектов);
- chess_pawn_black (1866 объектов);
- chess_pawn_white (1851 объектов);
- chess_queen_black (362 объектов);
- chess_queen_white (403 объектов);
- chess_rook_black (507 объектов);
- chess_rook_white (621 объектов).

Сильно выделяется большее количество объектов класса chess_pawn_black и chess_pawn_white, что в целом соответствует тому, что данных фигур в шахматной партии, как правило, большинство.

Раздельная разметка доски и фигур обеспечивает модульный подход к обучению моделей, позволяя независимо решать задачи сегментации игровой области и детектирования объектов.

Датасет содержит 1346 изображений, опубликован на huggingface [5]. На рис. 2 приведены примеры изображений.

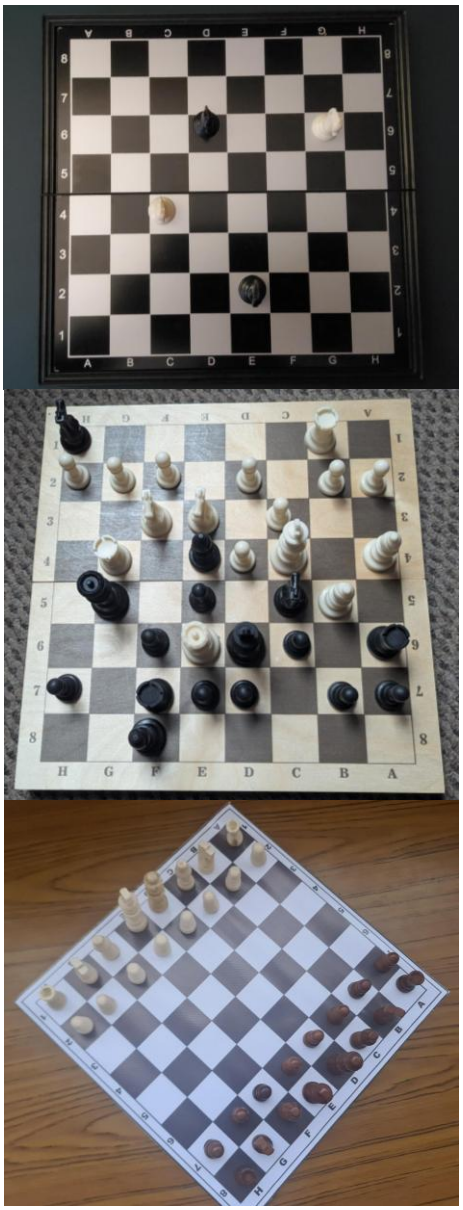


Рис. 2 Примеры изображений датасета с разным набором фигур

III. Сегментирование доски

Для выполнения задачи сегментирования шахматной доски использовалась модель yolov8n-seg. — это облегчённая модель для инстанс-сегментации, построенная на архитектуре YOLOv8 [6]. Она состоит из трёх основных частей: backbone, который извлекает признаки из изображения, neck, объединяющий признаки разных масштабов, и сегментационной головы, отвечающей за предсказание масок объектов (рисунок 3). В модели используются модули C2f, позволяющие уменьшить число параметров и ускорить работу сети. Сегментационная голова одновременно предсказывает ограничивающие рамки и пиксельные маски, что делает модель эффективной в решении задач сегментации в реальном времени. Благодаря компактной архитектуре

YOLOv8n-seg подходит для применения в системах с ограниченными вычислительными ресурсами.

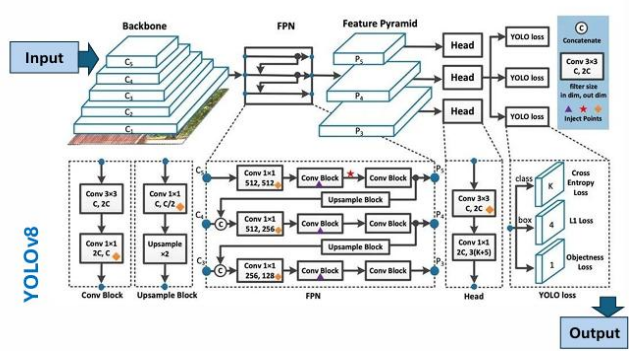


Рис. 3 Архитектура yolov8 [6]

Обучение сети производилось 50 эпох, размер батча 16. Рассматриваемые метрики: Precision (характеризует долю истинно положительных срабатываний среди всех обнаруженных объектов), Recall (измеряет способность модели находить все целевые объекты), mAP@50 (усредненное значение точности при пороге IOU 0.5) и mAP@50-95 (усредненное значение точности при порогах IOU от 0.5 до 0.95) [7]. Значения полученных метрик в таблице 1.

ТАБЛИЦА I. Результат обучения yolov8n-seg

Модель	Precision	Recall	mAP@50	mAP@50 -95
YOLOv8n-seg	0.98	0.99	0.95	0.95

Таким образом, yolov8n-seg с высокой точностью находит шахматную доску на изображении.

Результат сегментирования шахматной доски на изображении приведен на рисунке 4. Так как шахматная доска квадратной формы, то полученный контур аппроксимируется четырехугольником.

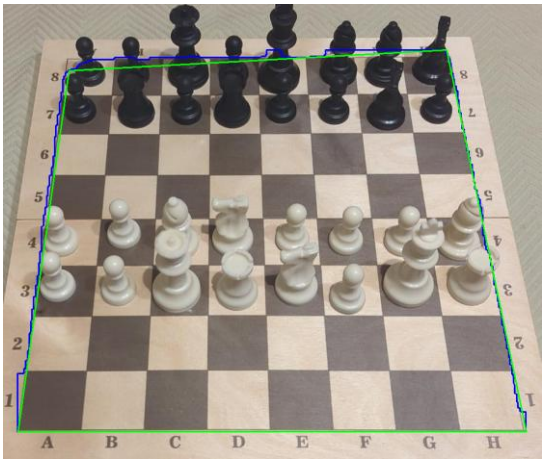


Рис. 4 Сегментирование шахматной доски. Синий – результат работы модели, зеленый – аппроксимация четырехугольником

Так как вершины полученного четырехугольника не всегда точно совпадают с границами доски, производится уточнение расположения доски средствами opencv [8], с помощью поиска положений пересечения клеток (рис. 5), на трансформированной доске (аппроксимирующий четырехугольник преобразуется в квадрат с вертикальными и горизонтальными сторонами рис. 6).

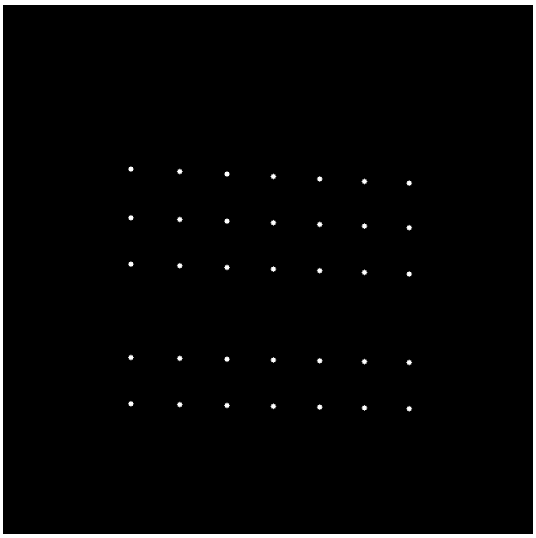


Рис. 5 Найденные пересечения клеток

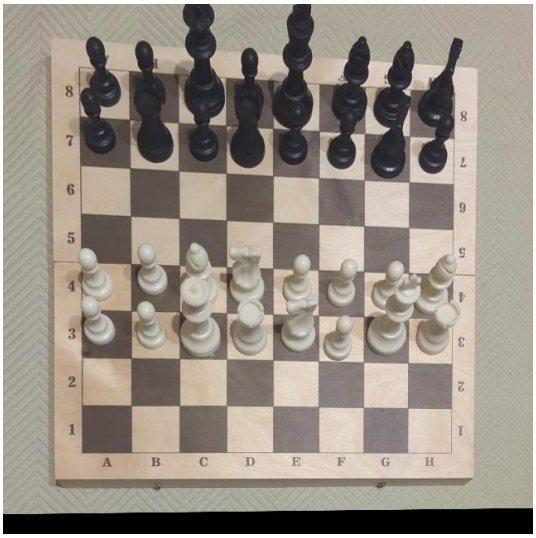


Рис. 6 Трансформированная шахматная доска

Некоторые пересечения обнаружить не удается (например, из-за перекрытия фигурами), но их можно восстановить по координатам смежных пересечений, что показано на рис. 7.

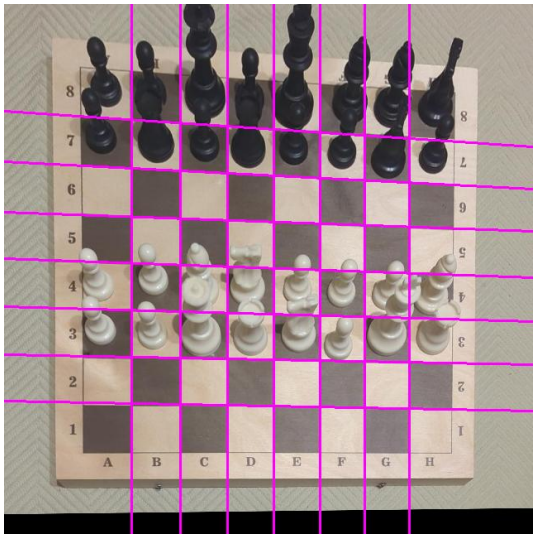


Рис. 7 Шахматная доска с найденными пересечениями клеток

IV. Детектирование фигур

A. *YOLO*

Алгоритмы семейства YOLO широко применяются в задачах детекции объектов в компьютерном зрении, например, в задаче детекции светофоров [9].

В работе применялись современные свёрточные и трансформерные архитектуры детектирования объектов для локализации и классификации шахматных фигур на изображениях. В качестве базовых моделей были выбраны компактные и эффективные версии алгоритмов семейства YOLO, учитывая их сочетание высокой скорости и приемлемой точности. Алгоритмы серии YOLO (You Only Look Once) — одностадийные CNN-детекторы с высокой скоростью инференса — активно используются в задачах реального времени.

Для исследования использовались облегчённые версии моделей — YOLO11n и YOLO12n, оптимизированные под высокоскоростное выполнение с минимальными задержками вывода. Архитектура YOLOv12, демонстрирует улучшенную способность к выделению локальных признаков при сохранении высокой скорости инференса, что критично для задач детекции множества мелких объектов на едином изображении [10,11].

Обучение производилось 200 эпох, размер батча 16. Матрица ошибок представлена на рисунке 8 и рисунке 9.

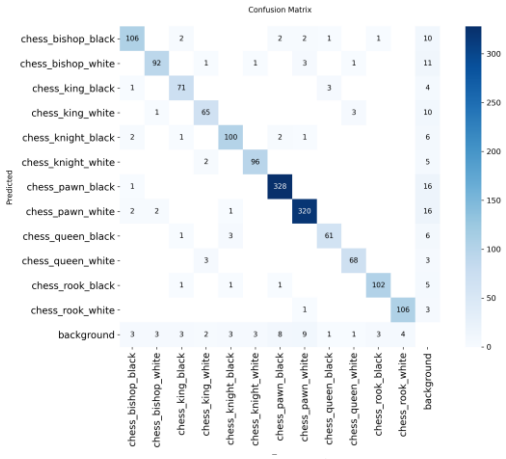


Рис. 8 Матрица ошибок YOLO11

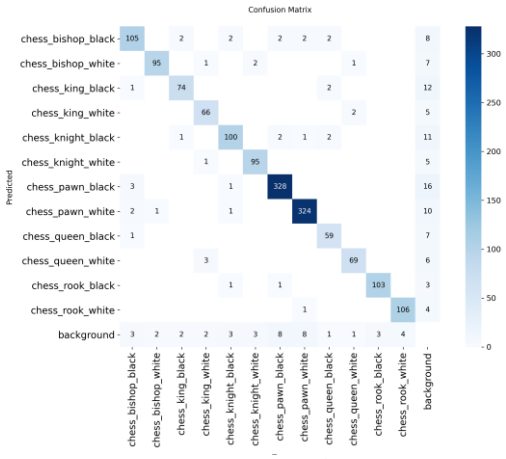


Рис. 9 Матрица ошибок YOLO12

B. *RF-DETR*

Помимо свёрточных моделей, в исследовании рассматривался трансформерный подход к детекции объектов на основе RF-DETR, являющийся развитием архитектуры DETR (DEtection TRansformer). В отличие от классических CNN-детекторов, трансформерные модели используют механизм самовнимания и глобальное представление сцены, что позволяет учитывать взаимное расположение объектов на изображении. Модель RF-DETR [12] оптимизирована с точки зрения архитектуры и вычислительной сложности, а её облегчённая версия (nano) демонстрирует конкурентоспособную точность при сниженных требованиях к вычислительным ресурсам.

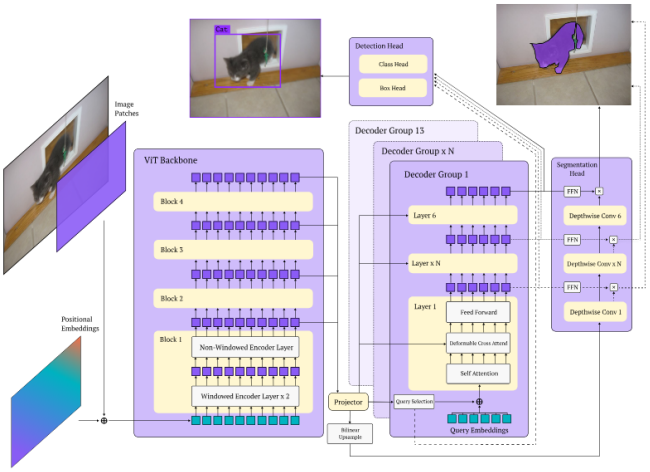


Рис. 10 Архитектура RF-DETR [12]

Обучение производилось 50 эпох, размер батча 4.

Все выбранные модели обучались на размеченном датасете изображений шахматных позиций с использованием прямоугольных аннотаций для фигур. В процессе инференса моделей YOLO применялись стандартные процедуры постобработки, включая порог уверенности и подавление немаксимумов, тогда как RF-DETR выполнял детекцию в рамках end-to-end трансформерной схемы сопоставления предсказаний с объектами.

Для сравнения моделей использовались метрики: Precision, Recall, mAP@50 и mAP@50-95.

ТАБЛИЦА II. Сравнение результатов обучения

	YOLO11	YOLOl2	RF-DETR
Precision	0.95	0.96	0.96
Recall	0.94	0.94	0.92
mAP@50	0.96	0.96	0.97
mAP@50-95	0.87	0.87	0.78

Из таблицы 2 следует, что модели семейства YOLO решают задачу лучше, исходя из метрики mAP@50-95. По другим метрикам результаты незначительно отличаются.

Используя результат сегментирования шахматной доски и области изображения, соответствующие фигуре каждого типа, можно восстановить позицию.

Помимо подхода, основанного на end-to-end детекции конкретных фигур, предлагается решение с детекцией любого объекта на шахматной доске и его дальнейшая классификация. Такой подход был предложен в статье [2], чтобы определить сигнал светофора. Для детектирования объектов на доске без классификации была предложена отдельная модель YOLO11n. Результаты ее обучения представлены на рисунке 11 и в таблице 3.

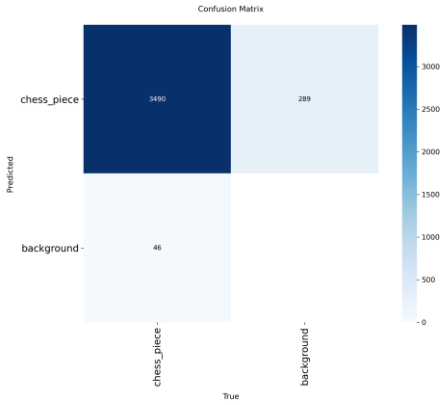


Рис. 11. Матрица ошибок YOLO11n на 1 классе

ТАБЛИЦА III. Результаты обучения на 1 классе

Модель	Precision	Recall	mAP@50	mAP@50 -95
YOLO11n	0.99	0.98	0.99	0.86

Итоговая модель демонстрирует более высокую точность в обнаружении фигур на доске и, согласно сравнительному анализу, приведенному в [13] для YOLO11n-obb с 15 классами и YOLO11n-pose с 1 классом, обладает теоретическим преимуществом в скорости обработки почти в два раза по сравнению с многоклассовыми версиями.

V. Классификация фигур

После детекции отдельных фигур на изображении каждая фигура вырезается по её ограничивающему прямоугольнику и подаётся на вход классификатору. Задача классификации заключается в определении одного из 12 классов. В рамках исследования были последовательно применены и проанализированы четыре современных подхода на основе глубокого обучения, каждый из которых демонстрирует различные принципы работы и стратегии обучения.

Для повышения устойчивости классификатора и борьбы с переобучением активно использовалась аугментация данных. В частности, применялись агрессивные трансформации, имитирующие сложные условия съёмки, такие как, изменение перспективы и геометрии объектов, а также обрезания изображений. Помимо этого, в рамках аугментации применялись базовые методы игры со светом, такие как случайное изменение яркости, контрастности и насыщенности. Эти простые приёмы позволили искусственно расширить обучающую выборку и улучшить обобщающую способность модели. Такой подход критически важен для подготовки классификатора к работе с неоднородными входными данными, так как он получает на вход вырезанные объекты, обработанные разными моделями детекции.

В качестве базовой свёрточной нейронной сети использовалась предобученная на наборе данных ImageNet архитектура ResNet-50, представленная в [14]. Её ключевая особенность — остаточные связи, которые решают проблему затухающего градиента и позволяют эффективно обучать очень глубокие сети. Для нашей

задачи модель была подвергнута дообучению: начальные слои, извлекающие общие признаки, были заморожены, а последние слои, включая классификационную голову, полностью переобучены на целевом датасете фигур.

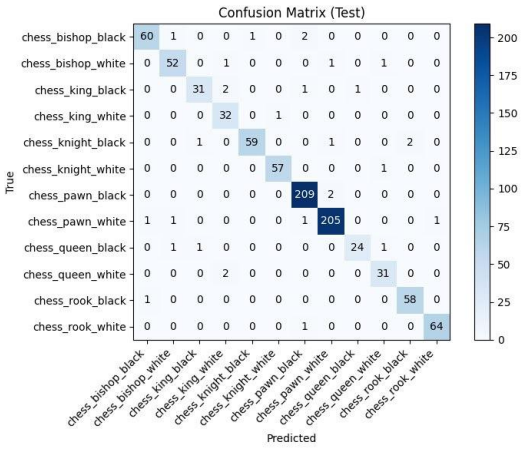


Рис. 12. Результат обучения ResNet-50 на тестовой выборке

Результат, представленный в виде матрицы ошибок на рисунке 12, показал итоговую точность в 94%, что подтвердило высокую эффективность глубоких CNN для задач классификации изображений после целевой адаптации [15].

Для оценки потенциала архитектур-трансформеров, доминирующих в обработке естественного языка, была использована модель Vision Transformer (ViT-Small/16), описанная исследователями в [16]. В отличие от индуктивно-свёрточных подходов, ViT разбивает входное изображение на неперекрывающиеся патчи размером 16×16 пикселей, которые затем линейно проецируются и обрабатываются механизмом само-внимания. Это позволяет модели улавливать глобальные контекстные зависимости между всеми частями изображения. Модель ViT-Small/16, предобученная на ImageNet-21k, была дообучена на наших данных с заменой классификатора.

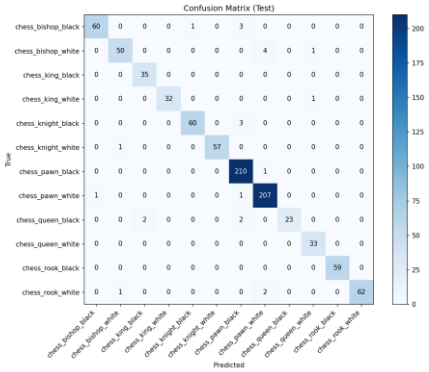


Рис. 13. Результат обучения ViT-Small/16 на тестовой выборке

Результат, представленный в виде матрицы ошибок на рисунке 13, продемонстрировал итоговую точность в 96%, что немного превысило показатель ResNet-50.

Современная CNN-архитектура ConvNeXt, предложенная в 2022 году в статье [17], представляет собой результат систематической модернизации классических ResNet-подобных сетей по принципам, заимствованным у ViT. Однако, в отличие от революционного подхода трансформерной модели, в которой свёртки заменяются механизмом внимания, ConvNeXt сохраняет чистую свёрточную основу, но радикально меняет её организацию и отдельные компоненты.

Архитектура включает несколько ключевых модификаций:

- Увеличение размера ядра свёртки до 7×7 для захвата большего контекста, подобно патчам в ViT;
- Замена активационной функции ReLU на более современный GELU;
- Применение послойной нормализации вместо пакетной.

Уменьшение количества активационных функций и нормализационных слоёв.

В результате модель сочетает вычислительную эффективность и индуктивные предпосылки CNN с мощным иерархическим представлением признаков, характерным для современных трансформеров.

В нашем эксперименте предобученная на ImageNet-22k модель ConvNeXt-Tiny была дообучена на целевом датасете фигур.

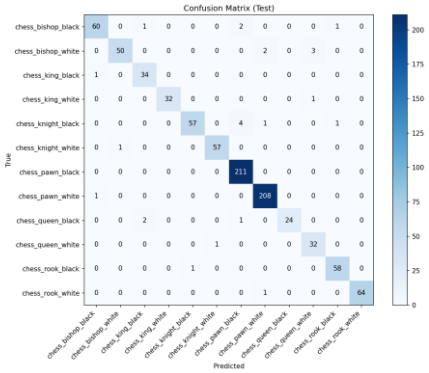


Рис. 14. Результат обучения ConvNeXt на тестовой выборке

На рисунке 14 представлена матрица ошибок. Точность итогового решения на тестовой выборке составляет 97%, что превосходит как классическую ResNet-50, так и ViT.

Особый интерес в рамках исследования представляет применение мультимодальных моделей, так как, в теории, они могли служить хорошим решением даже без обучения. В работе рассматривается модель CLIP, представленная в 2021 в статье [18]. В отличие от предыдущих архитектур, CLIP обучалась не на задачах классификации с фиксированным набором меток, а на контрастной задаче сопоставления произвольных пар «изображение–текст», собранных из интернета. Такое предобучение сформировало у модели обобщённые

семантические представления, связывающие визуальные паттерны с их текстовыми описаниями.

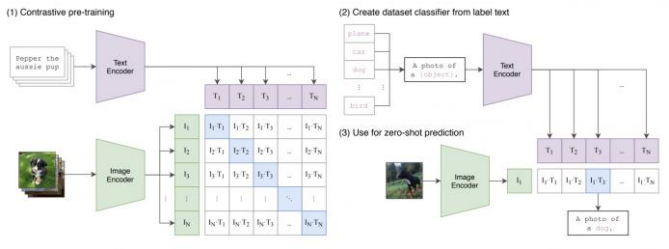


Рис. 15. Архитектура решений на CLIP [18]

На рисунке 15 представлена архитектура CLIP модели.

В первой части эксперимента был использован только vision-энкодер CLIP на архитектуре ViT-L/14, который был подвергнут стандартному дообучению для классификации 12 целевых классов фигур. Для этого его классификационная голова была заменена на новую, инициализированную случайным образом, и весь энкодер был дообучен на нашем датасете.

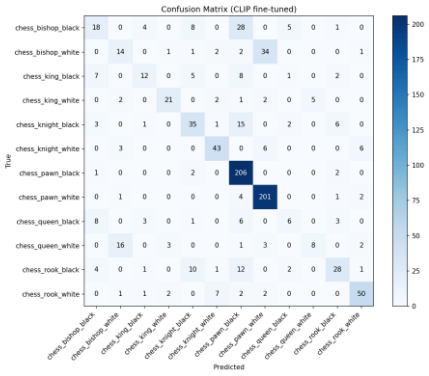


Рис. 16. Результат обучения CLIP (ViT-L/14) на тестовой выборке

Результат обучения представлен в виде матрицы ошибок на рисунке 16. Итоговая точность составила 71%, что значительно ниже показателей ResNet-50, ViT и ConvNeXt.

Помимо дообучения на целевом датасете фигур, не принесшего результатов, был проведён эксперимент с CLIP в режиме zero-shot классификации. Этот подход в полной мере раскрывает революционный потенциал модели. Классификация выполнялась путём сравнения эмбеддингов изображений и текстовых описаний классов в общем семантическом пространстве, созданном во время предобучения.

Так, для каждого из 12 классов были сформированы простые текстовые промпты по определенному шаблону. Эти текстовые строки кодировались текстовым энкодером CLIP. Каждое тестовое изображение кодировалось vision-энкодером. Далее для каждого изображения вычислялось косинусное сходство его вектора со всеми 12 текстовыми эмбеддингами. Присваивался тот класс, текстовый дескриптор которого оказался наиболее близок к изображению.

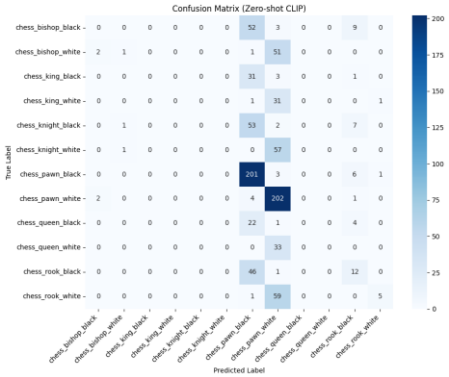


Рис. 17. Матрица ошибок CLIP на тестовой выборке

Результат обучения представлен в виде матрицы ошибок на рисунке 17. Точность zero-shot классификации составила всего 46%. Хотя этот результат количественно уступает всем дообученным моделям, его качественная значимость принципиально иная.

VI. Сравнение

На основе современных подходов компьютерного зрения для решения задачи детекции шахматных фигур на доске были реализованы и протестированы пять моделей: YOLO11, YOLO12, RF-DETR, YOLO+ViT и YOLO+ConvNeXt. Для предварительного обнаружения и кадрирования шахматной доски во всех экспериментах использовалась модель YOLOv8, выступающая в качестве унифицированного начального этапа обработки изображения.

После выполнения детекции модель формирует полную позицию: список фигур с их координатами и классами. На рисунке 18 представлено тестовое изображение, используемое для сравнительного анализа моделей. Рисунок 19 демонстрирует итоговые результаты распознавания, полученные каждым из подходов.

Рис. 18. Изображение для сравнения используемых моделей для получения позиции

а) б)

в) г)

д)

Рис. 19. 11 Результаты работы разных решений:
а) YOLO 11, б) YOLO 12, в) RF-DETR, г) YOLO+ViT, д) YOLO+ConvNeXt.

Проведенные эксперименты на реальных данных показали, что все модели успешно решают задачу детекции и классификации изолированных фигур при условии корректно определенной области доски. Основное узкое место всей системы выявилось на начальном этапе — в надежном поиске и сегментации шахматной доски на изображении. Изначальная идея создания системы, устойчивой к различным ракурсам съемки и условиям освещения, на практике столкнулась с проблемой: детектор доски часто не мог точно локализовать и разметить игровое поле для последующего перспективного преобразования, что влекло за собой полный сбой всего процесса распознавания.

Что касается классификации, здесь также были выявлены характерные ошибки:

- Для end-to-end детекторов (YOLO11, YOLO12, RF-DETR): Основные ошибки заключались в перепутывании визуально схожих фигур (например, ферзя и короля, особенно если они одного цвета и ракурс нечеткий);
- Для гибридных моделей с последующей классификацией (YOLO+ViT, YOLO+ConvNeXt): Проблемы были связаны с точным отделением пешек от фигур (особенно на сжатых или зашумленных изображениях после преобразования). Это может указывать как на недостаточную эффективность классификатора при работе с мелкими объектами, так и на проблемы дисбаланса классов в обучающей выборке, где пешки представлены в большем количестве, но в худшем качестве.

Еще одним важным критерием при сравнительном анализе предложенных подходов является скорость работы систем. Модели семейства YOLO (YOLO11 и YOLO12) являются самыми быстрыми в данном сравнении. Их архитектура, оптимизированная для end-to-end детекции и может достаточно быстро работать даже без видеокарты. Это делает их предпочтительными для сценариев, требующих работы в реальном времени или обработки больших объемов данных. RF-DETR работает не сильно медленнее YOLO.

На другом конце спектра находятся модели со сложными классификаторами, такие как YOLO+ViT и YOLO+ConvNeXt. Их вычислительная сложность значительно выше. Хотя начальный этап детекции фигур с помощью YOLO проходит быстро, но затем все обнаруженные области подаются на вход тяжелой классификационной модели ViT или ConvNeXt. Такие вычисления проходят быстро только при использовании видеокарты и с достаточной памятью для обработки всего батча.

VII. Заключение

В работе предложен и экспериментально оценён комплексный подход к автоматическому распознаванию шахматных позиций на изображениях, объединяющий современные методы компьютерного зрения, включая геометрическую сегментацию, детекцию объектов и глубокую классификацию. Создан размеченный датасет, охватывающий разнообразие стилей досок, фигур, ракурсов и условий освещения, что позволило провести репрезентативное сравнение современных архитектур.

Эксперименты показали, что наилучшее соотношение точности и скорости работы достигается при использовании end-to-end моделей семейства YOLO. Эти модели имеют высокую точность и способны работать в условиях реального времени даже на устройствах без GPU. Гибридные схемы обеспечили чуть более высокую классификационную точность, но более требовательны к вычислительным ресурсам и чувствительны к качеству детектирования объектов. А использование в такой схеме мультимодальной модели без обучения (в работе рассматривалась CLIP) оказалось непрактичным для решения поставленной задачи.

Особое внимание было уделено этапу локализации доски, где YOLOv8n-seg показала превосходные метрики, однако дальнейшая устойчивость всей системы оказалась критически зависимой от качества перспективного преобразования — это выявило основное узкое место в алгоритме: даже высокоточная

детекция фигур теряет смысл, если доска выделена неточно.

Для дальнейшего улучшения предложенного решения можно решить следующие задачи:

1. Разработка модели для точного определения ориентации и перспективы доски.

2. Адаптация системы к мобильным платформам. Для практического внедрения (например, в смартфоны или шахматные часы с камерой) необходимо обеспечить задержку <200 мс на устройствах среднего класса.

3. Расширение датасета с помощью генеративных и RAG-подобных методов. В частности, комбинирование реальных изображений досок с синтетическими фигурами, размещёнными с учётом физики освещения и теней, по аналогии с подходом, описанным в статье [19]. Такой гибридный синтез снизит зависимость от дорогостоящей ручной разметки.

Литература

[1] Солодов, С. В. Методы исследования дефектов дорожного покрытия в видеопотоке камер портативных устройств / С. В. Солодов, С. В. Проничкин // Новые материалы и перспективные технологии : СБОРНИК МАТЕРИАЛОВ ПЯТОГО МЕЖДИСЦИПЛИНАРНОГО НАУЧНОГО ФОРУМА С МЕЖДУНАРОДНЫМ УЧАСТИЕМ, Москва, 30 октября – 01 2019 года. Том I. – Москва: Интеллектуальные системы, 2019. – С. 418-421. – EDN SZLIDC.

[2] Использование систем технического зрения для распознавания стрелочных переводов в задаче навигации подвижного рельсового состава / Р. Р. Бикмаев, А. В. Попов, Р. Н. Садеков [и др.] // Труды ФГУП "НПЦАП". Системы и приборы управления. – 2018. – № 1. – С. 43-44. – EDN UYKJYL.

[3] Катызина, А. А. Распознавание рукописных математических выражений с использованием нейронных сетей / А. А. Катызина, А. Т. Фам // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : Сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 30 мая 2025 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2025. – С. 63-69. – EDN HPDCOA.

[4] Czyzewski, Maciej & Laskowski, Artur & Wasik, Szymon. (2020). Chessboard and Chess Piece Recognition With the Support of Neural Networks. Foundations of Computing and Decision Sciences. 45. 257-280. 10.2478/fcds-2020-0014.

[5] Huggingface — Chess Dataset. – 2025 – Available at: https://huggingface.co/datasets/pashawh4/chess (Accessed: 25.12.2025)

[6] Zhang Y., Li H., Wang Q. YOLOv8-seg-CP: An Improved YOLOv8 Segmentation Model for Complex Scene Perception // Scientific Reports. – 2024. – Vol. 14. – Article number: 78578.

[7] Базалеев, Ф. Е. Исследование возможности детектирования дорожных знаков на основе нейросетевой модели YOLO / Ф. Е. Базалеев, Е. И. Пиховская // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры "Инженерной кибернетики", Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 29-33. – EDN FZGQPG.

[8] OpenCV. Open Source Computer Vision Library (Version 4.11.0) [Электронный ресурс]. URL: https://opencv.org (дата обращения: 25.12.2025)

[9] Анализ нейронных сетей для детектирования светофоров на изображениях / Л. С. Толстенко, А. А. Клейменов, Б. Али [и др.] // Известия Института инженерной физики. – 2023. – № 2(68). – С. 59-65. – EDN VRTWMF.

[10] 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.

[11] Wang C.-Y., Bochkovskiy A., Liao H.-Y. M. YOLOv12: Attention-Centric Real-Time Object Detectors // arXiv preprint. – 2025. – arXiv:2502.12524.

[12] Roboflow Research Team. RF-DETR: Architecture-Optimized Detection Transformer for Real-Time Object Detection // arXiv preprint. – 2024. – arXiv:2411.09554.

[13] Jocher G., Qiu J. Ultralytics YOLO11 (Version 11.0.0) [Программное обеспечение]. – 2024. – URL: https://github.com/ultralytics/ultralytics (дата обращения: 25.12.2025).

[14] He K., Zhang X., Ren S., Sun J. Deep Residual Learning for Image Recognition // 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). – 2016. – P. 770–778.

[15] Kornblith S., Shlens J., Le Q. V. Do better ImageNet models transfer better? // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). – 2019. – P. 2661-2671.

[16] Dosovitskiy A., Beyer L., Kolesnikov A., et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale // International Conference on Learning Representations (ICLR). – 2021.

[17] Liu Z., Mao H., Wu C.-Y., et al. A ConvNet for the 2020s // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). – 2022. – P. 11976–11986.

[18] Radford A., Kim J. W., Hallacy C., et al. Learning Transferable Visual Models From Natural Language Supervision // Proceedings of the 38th International Conference on Machine Learning (ICML). – 2021. – Vol. 139. – P. 8748–8763.

[19] Семенов, И. А. Генеративная нейросетевая модель для аугментации изображений двумерных штрихкодов / И. А. Семенов // Труды 67-й Всероссийской научной конференции МФТИ, 31 марта – 5 апреля 2025 г. Прикладная математика и информатика. – М: Физматкнига, 2025. – С. 226–227.

ГРНТИ 28.23.15
ASJC 1707
OECD 1.02 EP

Исследование эффективности современных нейросетевых архитектур для классификации твердых бытовых отходов

Д. В. Чун
кафедра инженерной кибернетики
НИТУ «МИСИС»
Москва, Россия
m2103009@edu.misis.ru

***Аннотация*— в работе рассматривается концепция автоматизации процесса сортировки отходов с применением технологий компьютерного зрения. Исследование направлено на решение задачи классификации мусора по пяти основным категориям: картон, стекло, металл, бумага и пластик. В качестве основного механизма распознавания используются современные нейросетевые архитектуры. Особое внимание уделено устойчивости моделей к деформациям объектов и изменениям условий освещения, поскольку данный аспект является особенно важным в промышленных условиях. В ходе исследования проводится сравнительный анализ нейронных сетей и визуальных трансформеров в условиях перехода от синтетических обучающих выборок к реальным сценариям эксплуатации. Итогом работы является выявление наиболее устойчивой к различным условиям архитектуры.**

Ключевые слова — глубокое обучение, классификация мусора, компьютерное зрение, нейронные сети, распознавание образов.

I. Введение

Рост объема бытовых отходов является одной из глобальных экологических проблем на данный момент. Традиционные методы механической и ручной сортировки мусора на перерабатывающих предприятиях обладают рядом ограничений, включая низкую скорость обработки потока и потенциальные риски для здоровья персонала. Использование систем с использованием моделей глубокого обучения позволит автоматизировать процесс идентификации типов материалов, что критически важно для повышения доли вторичного использования ресурсов [1, 2].

Данная работа предлагает исследование применимости моделей класса State-of-the-Art (SOTA) к визуальным данным, характеризующимся высокой степенью зашумленности. В отличие от стандартных задач распознавания образов, классификация мусора осложняется сильными деформациями объектов (смятые бутылки и коробки), наличием загрязнений, перекрытием этикетками и вариативностью фона [3,4], что сильно затрудняет обработку и присвоение типа отдельным объектам. Также важно учитывать разнообразие объектов внутри одного класса. Один материал может иметь разное применение и внешний вид, что может помешать корректной работе модели [5,6].

Основная цель исследования – провести сравнение актуальных архитектур нейронных сетей и оценить их способность к обобщению признаков.

II. Наборы данных

Для достижения высокой точности распознавания был разработан композитный датасет, суммарный объем которого составил 16500 изображений. Процесс формирования был разделен на несколько этапов.

A. Сбор и первичная обработка

Первоначально был создан набор данных из 1500 объектов, собранный с использованием поисковых систем Bing и Yandex. Для обеспечения высокого качества обучения использовались уточняющие запросы (например, «crushed plastic bottle», «dirty waste cardboard»), позволяющие получить изображения объектов в состояниях, близких к реальным отходам. Также для обеспечения устойчивости модели к различным условиям была проведена аугментация. Однако предварительные эксперименты показали низкую точность на реальных фото.

B. Обогащение и композитное объединение

Для расширения пространства признаков вышеупомянутая выборка была дополнена эталонными изображениями из открытых репозиториев, включая Garbage Classification Dataset [7]. В ходе объединения был проведен маппинг классов, в частности, различные категории стекла (зеленое, коричневое, прозрачное) были объединены в единый класс glass. Итоговый массив данных был распределен по 5 категориям:

- Cardboard (картон): упаковочные коробки, гофрокартон, яичные лотки;
- Glass (стекло): целые и битые бутылки, банки;
- Metal (металл): алюминиевые банки, фольга, крышки;
- Paper (бумага): газетная продукция, офисная бумага, листовки, чеки;
- Plastic (пластик): ПЭТ-бутылки, ПНД-канистры, полиэтиленовые пакеты, контейнеры.

C. *Аннотирование в среде CVAT.AI*

Для превращения собранного массива изображений в полноценный обучающий набор данных была проведена процедура аннотирования. В рамках проекта отобранные экземпляры были выгружены, и каждому из них была присвоена одна из пяти категориальных меток. Таким образом была обеспечена корректная структура данных для последующего экспорта в форматы, совместимые с современными фреймворками глубокого обучения. На рисунке 1 продемонстрирован интерфейс системы в процессе разметки категории «Cardboard».

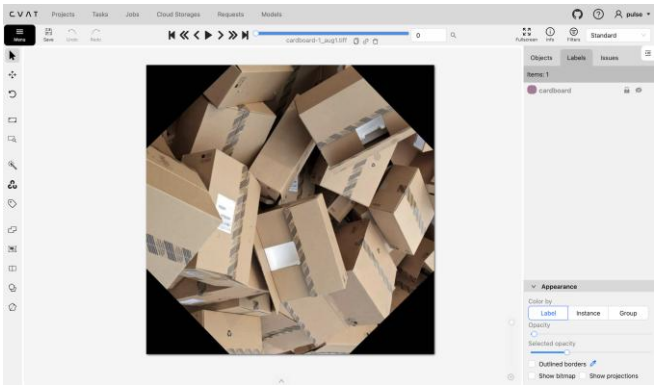


Рисунок 1 – Процесс аннотирования и присвоения категориальных метод изображениям в интерфейсе CVAT AI

Сформированный датасет опубликован в открытом доступе на платформе Hugging Face [8].

III. Нейросетевые архитектуры

В исследовании проведено сравнение архитектур, представляющих три различных подхода к анализу изображений: модернизированные сверточные сети, мобильные архитектуры и трансформеры.

A. *ConvNeXt V2 (Nano)*

Модель ConvNeXt V2 [9] представляет собой современную итерацию сверточных сетей, переосмысленную под влиянием Vision Transformers. В архитектуру внедрена технология Global Response Normalization (GRN), которая повышает конкуренцию признаков между солями, и используется метод предобучения Masked Autoencoder (MAE). Это позволяет сети эффективно «достраивать» структуру объекта, даже если часть его скрыта или деформирована [10]. Схема блока приведена на рисунке 2

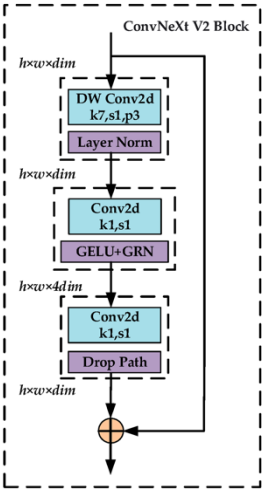


Рисунок 2 – Схематическое описание внутреннего блока архитектуры ConvNeXt V2 [11]

B. *MobileNetV4 (Medium)*

MobileNetV4 [12] является новейшим представителем семейства эффективных моделей, оптимизированных для мобильных платформ. В основе лежит блок Universal Inverted Bottleneck (UIB), который позволяет динамически адаптировать вычислительный граф под конкретное аппаратное обеспечение. В данной модели минимизировано использование тяжелых операций, что обеспечивает высокую пропускную способность при малом количестве параметров. Структура слоев MobileNetV4 представлена на рисунке 3.

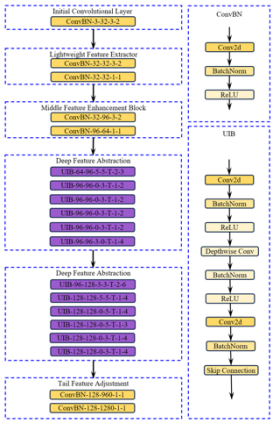


Рисунок 3 – Организация слоев в блоке Universal inverted Bottleneck архитектуры MobileNetV4 [13]

C. *Swin Transformer V1 (Tiny)*

Swin Transformer [14] реализует иерархический подход к построению визуальных представлений. В отличие от стандартных Vision Transformers, Swin вычисляет самовнимание в неперекрывающихся локальных окнах, которые сдвигаются (shifted windows) между слоями. Это обеспечивает линейную сложность вычислений относительно размера изображения и позволяет модели эффективно улавливать пространственные связи объектов мусора. Схема разбиения на окна приведена на рисунке 4.

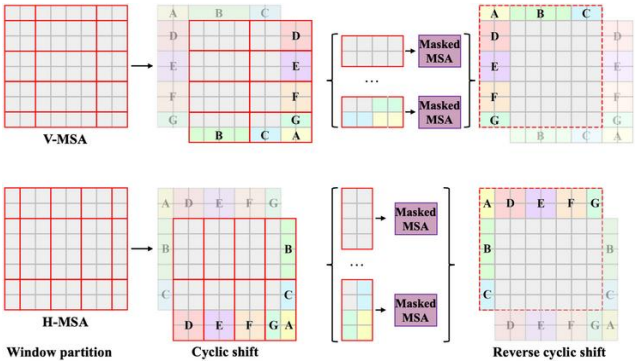


Рисунок 4 – Механизм смещенных окон в архитектуре Swin Transformer [15]

D. EVA-02 (Tiny)

EVA-02 [16] — это передовая архитектура, ориентированная на достижение максимальной точности за счет использования маскированного моделирования изображений (MIM). Модель обучается восстанавливать скрытые фрагменты изображений, что развивает глубокое понимание геометрии объектов. Архитектура базируется на оптимизированных блоках внимания с улучшенной инициализацией весов. Схематическое представление архитектуры EVA-02 приведено на рисунке 5.

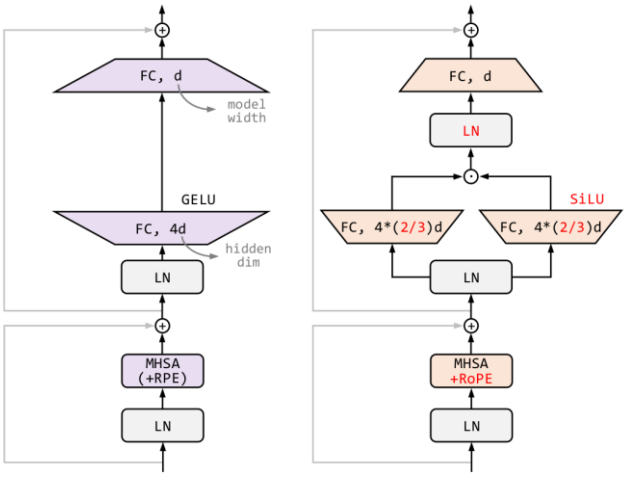


Рисунок 5 – Общая схема архитектуры EVA-02 на базе Vision Transformer [17]

IV. Сравнение

Обучение проходило в течение 20 эпох. Использовался оптимизатор AdamW с начальной скоростью обучения 0.0001 и весовой нормализацией [18]. Перед началом анализа были сформулированы метрики эффективности. Они представлены в таблице 1.

ТАБЛИЦА I. Параметры сравнения эффективности моделей

	Описание	**Единица измерения**
Точность (Accuracy)	Основной показатель качества классификации. Отражает долю объектов мусора, чей материал был определен верно	%
Пропускная способность (FPS)	Скорость обработки видеопотока. Критически важна для работы автоматизированных конвейерных лент в реальном времени	Кадр/с
Объем параметров (M)	Показатель вычислительной сложности и веса модели. Определяет требования к памяти устройства	Млн
Задержка (Latency)	Время, затрачиваемое на инференс одного изображения. Характеризует отзывчивость системы при одиночных запросах	мс

A. Результаты на валидационной выборке

Первый этап оценки проводился на отложенной валидационной выборке. Результаты представлены в таблице 2. Результаты тестирования моделей на валидационной выборке

ТАБЛИЦА II. Показатели эффективности моделей на валидационной выборке

Модель	**Accuracy**	**M**	**Latency (мс)**	**FPS**
Swin V1	100%	27.5	34.2	29.2
ConvNeXt V2	99.7%	15.0	33.3	30.0
EVA-02	99%	5.5	34.6	28.9
MobileNetV4	97.7%	8.4	36.9	27.1

Как видно из таблицы, в данных условиях все архитектуры демонстрируют высокую точность. Модель ConvNeXt V2 показала лучшую производительность (30 FPS) и минимальную задержку (33.3 мс), опередив легковесную MobileNetV4. Swin достиг 100% точности, однако это может свидетельствовать о переобучении.

B. Результаты на валидационной выборке

Ключевым этапом исследования стала проверка моделей на независимом тестовом наборе, содержащем сложные объекты с нестандартной геометрией и текстурой. Результаты тестирования приведены в таблице 3.

ТАБЛИЦА III. Точность классификации на независимой тестовой выборке

Модель	**Accuracy**	**M**	**Latency (мс)**	**FPS**
Swin V1	68%	27.5	15.9	62.7
ConvNeXt V2	80%	15.0	19.1	52.4
EVA-02	64%	5.5	12.76	78.4
MobileNetV4	76%	8.4	15.5	64.4

На основе данной таблицы можно сделать следующие выводы:

- Точность классификации: безусловным лидером является сверточная нейронная сеть

ConvNeXt V2, достигшая точности 80% на реальных данных. Она продемонстрировала наилучшую устойчивость к вариативности объектов. Модель MobileNetV4 показала достойный результат (76%), незначительно уступив лидеру. В свою очередь трансформерные архитектуры оказались менее эффективными в данной задаче в силу ограниченности обучающей выборки;

- Скоростные показатели: самой быстрой архитектурой оказалась EVA-02, обрабатывающая 78.4 кадра в секунду. Однако низкая точность делает ее использование менее привлекательным;
- Баланс характеристик: оптимальное соотношение иллюстрирует MobileNetV4. При высокой скорости она обеспечивает качество работы, близкое к лидеру. Тем не менее для систем сортировки, где приоритетом является чистота фракций, предпочтительным выбором также остается ConvNext V2. Несмотря на наименьший показатель FPS данной модели, ее скорость более чем достаточно для работы в режиме реального времени в стандартных условиях, где пропускная способность промышленных камер – 30 FPS.

V. Заключение

В ходе работы было проведено комплексное исследование эффективности современных нейросетевых архитектур в задаче классификации бытовых отходов. Эксперименты продемонстрировали преимущество современных сверточных сетей над визуальными трансформерами в условиях ограниченной и неоднородной выборки. Модель архитектуры ConvNext продемонстрировала более высокую способность к обобщению текстурных признаков материалов, что позволило минимизировать ошибки распознавания на сложных объектах с деформациями и бликами.

Анализ соотношения скорости и качества показал, что рекордная производительность сопровождается снижением точности, что ограничивает применение наиболее быстрых моделей в автономных системах сортировки, где критична чистота разделения фракций. Однако архитектура MobileNetV4 подтвердила эффективность для сценариев с ограниченными вычислительными ресурсами, обеспечив компромисс между метриками и потеряв лишь 4% точности.

Таким образом, для реализации стационарных комплексов для автоматизированной сортировки наиболее перспективной была признана ConvNext V2. Несмотря на более высокие требования к вычислительным мощностям по сравнению с легковесными аналогами, данная архитектура обеспечивает необходимый запас скорости, достаточный для обработки видеопотока.

Литература

[1] J. Bobulski and M. Kubanek, “Deep learning for plastic waste classification system”, Applied Computational Intelligence and Soft Computing, vol. 2021, 2021.

[2] Елисеев, А. Н. Решение задачи классификации сельскохозяйственных культур с использованием глубоких нейронных сетей / А. Н. Елисеев, И. И. Курочкин // Облачные и распределенные вычислительные системы в электронном управлении. ОРВС - 2023 : сборник трудов 4-й международной научно-технической конференции. — Курск: ЗАО «Университетская книга», 2024. — С. 35-40.

[3] T. J. Sheng et al., “An intelligent waste classification system using deep learning with a novel dataset”, IEEE Access, vol. 8, pp. 171195–171206, 2020.

[4] Криворот, Ю. А. Классификация ядовитых и неядовитых видов грибов / Ю. А. Криворот, С. С. Белякова // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики», Москва, 30 мая 2025 года. — Москва: Национальный исследовательский технологический университет «МИСИС», 2025. — С. 83-88.

[5] Заречнев, Д. В. Классификация спектров растений методами машинного обучения / Д. В. Заречнев, И. И. Курочкин // Облачные и распределенные вычислительные системы в электронном управлении. ОРВС - 2023 : сборник трудов 4-й международной научно-технической конференции. — Курск: ЗАО «Университетская книга», 2024. — С. 41-43.

[6] Солодов, С. В. Методы исследования дефектов дорожного покрытия в видеопотоке камер портативных устройств / С. В. Солодов, С. В. Проничкин // Новые материалы и перспективные технологии : сборник материалов пятого междисциплинарного научного форума с международным участием, Москва, 30 октября — 01 ноября 2019 года. — Москва: Интеллектуальные системы, 2019. — Т. I. — С. 418-421.

[7] Asdasdasasdas, “Garbage Classification Dataset (12 classes)”, Kaggle Repository. Available at: https://www.kaggle.com/datasets/asdasdasasdas/garbage-classification (Accessed: December 15, 2025).

[8] Skyaker, “Garbage Collection Dataset”, Hugging Face Repository. Available at: https://huggingface.co/datasets/skyaker/garbage-collection (Accessed: December 24, 2025).

[9] S. Woo et al., “ConvNeXt V2: Co-designing and Scaling ConvNets with Masked Autoencoders”, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 16133–16142.

[10] Панкратов, А. Р. Классификация болезней томатов при помощи компьютерного зрения / А. Р. Панкратов, Т. В. Конев // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики», Москва, 26–27 декабря 2024 года. — Москва: Национальный исследовательский технологический университет «МИСИС», 2024. — С. 110-115.

[11] Y. Li et al., “Lightweight Algorithm for Rail Fastener Status Detection Based on YOLOv8n”, Electronics, vol. 13, no. 17, 2024.

[12] D. Qin et al., “MobileNetV4: Universal Models for the Mobile Ecosystem”, arXiv preprint arXiv:2404.10518, 2024.

[13] Y. Liu, X. Han, J. Wang, and Y. Wang, “YOLOv8-MSP-PD: A Lightweight YOLOv8-Based Detection Method for Jinxiu Malus Fruit in Field Conditions”, Agronomy, vol. 15, no. 7, 2025.

[14] Z. Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”, in Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 10012–10022.

[15] A. Brăteanu et al., “Pre-trained low-light image enhancement transformer”, IET Digital Library. Available at: https://www.researchgate.net/publication/378900397 (Accessed: December 24, 2025).

[16] Y. Fang et al., “EVA-02: A Visual Representation for Neon Genesis”, arXiv preprint arXiv:2303.11331, 2023.

[17] MMPretrain Contributors, “EVA-02 — MMPretrain documentation”. Available at: https://mmpretrain.readthedocs.io/en/dev/papers/eva02.html (Accessed: December 16, 2025).

[18] I. Loshchilov and F. Hutter, “Decoupled Weight Decay Regularization”, in International Conference on Learning Representations (ICLR), 2019.

ГРНТИ 28.23.15, 28.23.37
ASJC 1707
OECD 01.02.EP

Эмбединги как инструмент для анализа визуального сходства

Г. В. Шевченко
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2510747@edu.misis.ru

***Аннотация* — в данной работе рассматривается задача межмодального поиска и анализа визуально-текстового сходства на основе мультимодальных векторных представлений. Основное внимание уделяется исследованию свойств эмбеддингов, формируемых предобученными моделями типа CLIP, и их способности отражать семантическое соответствие между изображениями и текстовыми описаниями. В рамках исследования анализируется, насколько расстояния между эмбеддингами в латентном пространстве коррелируют с корректностью межмодального сопоставления. Экспериментальная оценка проводится на собственном размеченном датасете с использованием метрики Recall@K. Полученные результаты показывают, что мультимодальные эмбеддинги CLIP обеспечивают более точное согласование визуальных и текстовых представлений по сравнению с традиционными архитектурами с раздельными энкодерами.**

Ключевые слова — CLIP, Recall@K, визуально-текстовое сходство, векторные представления (эмбеддинги), межмодальный поиск, мультимодальные модели.

I. Введение

В последние годы задачи автоматической оценки визуального сходства изображений приобретают всё большую актуальность в контексте развития систем контент-поиска, рекомендательных сервисов и управления цифровыми коллекциями. Однако существующие подходы, основанные на ручном выделении признаков или даже глубоких нейронных сетях, часто оказываются недостаточно гибкими для учета семантической схожести, выходящей за рамки жестко заданных категорий. Это создает потребность в методах, способных формализовать субъективное визуальное сходство на основе семантически насыщенных векторных представлений.

Классические подходы компьютерного зрения, опирающиеся на ручной инженерный выбор признаков (например, SIFT, цветовые гистограммы) или даже обученные сверточные нейронные сети для классификации, часто оказываются ограниченными в задачах тонкого сравнения, выходящего за рамки базовых категорий. Эти методы могут не учитывать комплексные семантические аспекты, определяющие восприятие сходства.

Перспективным направлением является использование мультимодальных моделей, таких как CLIP[1], которые обучаются на согласовании изображений и текстовых описаний. Такие модели формируют общее латентное пространство, в котором близость векторов отражает семантическую связь между модальностями [2]

Целью данной работы является оценка эффективности мультимодальных эмбеддингов модели CLIP для задачи межмодального поиска изображений и текстовых описаний.В рамках работы проводится сравнение CLIP с архитектурой ResNet50 + BERT на собственном размеченном датасете с использованием метрики Recall@K.

II. Наборы данных

В данной работе для проведения экспериментального анализа использовались наборы данных различного происхождения. Обучение нейросетевых моделей в рамках исследования не выполнялось — все эксперименты проводились с использованием предобученных архитектур. Наборы данных применялись исключительно для оценки качества формируемых эмбеддингов и анализа визуального сходства изображений.

A. Flickr30k

Flickr30k [4] является стандартным мультимодальным датасетом, широко используемым для задач связывания изображений и текста (image-text retrieval), генерации описаний (image captioning) и оценки качества мультимодальных эмбеддингов. Набор данных основан на коллекции из 31 783 фотографий, отобранных из публичного онлайн-сервиса Flickr.com с акцентом на разнообразие повседневных человеческих активностей, сцен, объектов и событий.

Визуальные особенности и состав. Изображения в датасете представляют собой фотографии, сделанные непрофессиональными пользователями в естественных, непостановочных условиях. Как показано на рисунке 1, это обеспечивает высокую вариативность по ключевым параметрам:

- Необычные углы: использование перевернутого кадра для создания эффекта дезориентации (фото а).
- Планы: от выразительного крупного плана (портрет в шляпе на фото б) до общих планов, вписывающих человека в архитектуру здания или масштаб улицы (фото в, г).
- Динамика: фиксация движения в прыжке, создающая ощущение «замершего времени» (фото д).

Рис 1. Примеры фотографий из датасета

B. *Сборка сообственного датасета*

Для исследования того, как векторные представления (эмбеддинги) кодируют и отражают визуальное сходство изображений, была сформирована небольшая вручную размеченная выборка изображений. Формирование выборки проводилось с ориентацией на принципы, используемые в задачах оценки визуального сходства и межмодального поиска, где важным является наличие вариативных и частично пересекающихся визуальных признаков между различными изображениями.

Исходный материал был собран путём целенаправленного отбора изображений, отражающих объекты и сцены с выраженными атрибутами формы, текстуры, цвета и композиции. В выборку включались изображения, снятые в различных условиях освещения, а также в разных контекстах, что обеспечило вариативность визуальных характеристик внутри семантически близких групп. Общий объём исходных изображений составил около 100 экземпляров[17].

Для обеспечения корректности последующего анализа изображения были приведены к унифицированному формату и разрешению, а также подвергнуты минимальной предварительной обработке. Такой подход позволил исключить влияние технических различий входных данных на формирование эмбеддингов.

С целью повышения разнообразия визуальных сценариев в выборку были включены как простые случаи очевидного сходства, так и более сложные примеры, где сходство носит контекстуальный характер или проявляется лишь в отдельных визуальных аспектах. Это позволило оценить устойчивость эмбеддингов к вариациям, не влияющим на общее семантическое содержание сцены.

Аннотирование выполнялось для задачи связывания изображений с текстовыми описаниями (image–text retrieval). Каждому изображению вручную присваивалось текстовое описание на английском языке, отражающее основные объекты, действия и контекст сцены. Среднее время разметки одного изображения составляло порядка 1–3 минут.

Данная выборка не разделялась на обучающую, валидационную и тестовую части и не использовалась для обучения или дообучения моделей. Она применялась исключительно в качестве оценочного набора для сравнения качества эмбеддингов, формируемых различными предобученными архитектурами. Размеченная выборка была опубликована в открытом доступе на платформе HuggingFace [17].

Рис 2. Пример создания меток в CVAT.AI

III. Нейросетевые архитектуры

A. *Модель CLIP*

CLIP (Contrastive Language-Image Pre-training) - это новаторская архитектура мультимодальных моделей, представленная OpenAI, которая преодолевает разрыв между компьютерным зрением и обработкой естественного языка обработки. В отличие от традиционных систем компьютерного зрения, обучающихся на фиксированных наборах предварительно помеченных категорий, CLIP[5] учится ассоциировать изображения с текстовыми описаниями, тренируясь на сотнях миллионов пар "изображение-текст", собранных из интернета. интернета. Такой подход позволяет модели понимать визуальные концепции через призму естественного языка, что дает возможность, известную как обучение с нулевым результатом, когда модель может правильно классифицировать изображения по категориям, которые она никогда явно не видела во время обучения. Благодаря согласованию визуальной и текстовой информации в общем пространстве признаков, CLIP служит универсальной базовой моделью для широкого спектра последующих задач искусственного интеллекта задач, чтобы максимизировать косинусоидальное сходство между вкраплениями правильных пар и минимизировать сходство для неправильных пар

B. *Архитектура модели CLIP*

Метод контрастивного языково-изображенного предварительного обучения (CLIP) использует архитектуру с двумя кодировщиками для отображения изображений и текста в общее пространство. Он работает путем совместного обучения двух кодировщиков: одного для изображений (Vision Transformer)[7] и одного для текста (языковая модель на основе Transformer)

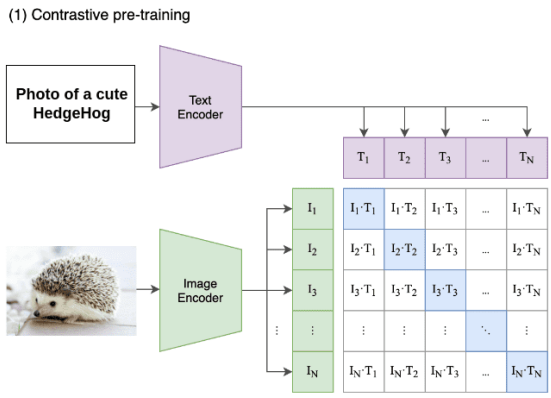


Рисунок 3. Архитектура модели CLIP

- Кодировщик изображений: Кодировщик изображений извлекает важные признаки из визуального входного изображения. Этот кодировщик принимает «изображение в качестве входных данных» и создает многомерное векторное представление [16]. Обычно он использует архитектуру сверточной нейронной сети (CNN), например ResNet [9], для извлечения признаков изображения.
- Кодировщик текста: Кодировщик текста кодирует семантическое значение соответствующего текстового описания. Он принимает на вход текстовую подпись/метку и создает другое многомерное векторное представление. Часто для обработки текстовых последовательностей используется архитектура на основе трансформеров, например, Transformer или BERT[10].
- Общее пространство встраивания: Два кодировщика создают встраивания в общем векторном пространстве. Эти общие пространства встраивания позволяют CLIP сравнивать текстовые и графические представления и изучать их взаимосвязи.

C. Процесс обучения модели CLIP

CLIP предварительно обучается на крупномасштабном наборе данных, содержащем 400 миллионов изображений и текстовых данных, собранных из интернета. В процессе предварительного обучения модели предоставляются пары изображений и текстовых подписей. Некоторые из этих пар являются подлинными совпадениями, в то время как другие не совпадают. Создаются общие эмбеддинги пространства.

Для каждого изображения создается несколько текстовых описаний, включая правильное и несколько неправильных. Это приводит к образованию смеси положительных (совпадающих) и отрицательных (несовпадающих) пар. Эти описания передаются в текстовый кодировщик, генерируя векторные представления, специфичные для каждого класса [6].

На этом этапе в игру вступила еще одна важная функция: функция контрастных потерь. Эта функция штрафует модель за неправильное сопоставление пар (изображение-текст), но вознаграждает ее за правильное сопоставление пар (изображение-текст) в латентном пространстве. Она побуждает модель изучать представления, которые точно отражают сходство визуальной и текстовой информации.

Теперь обученный текстовый кодировщик используется в качестве классификатора с нулевым уровнем предварительного обучения. С новым изображением CLIP может делать прогнозы без предварительного предварительного обучения. Это достигается путем пропуска изображения через кодировщик и классификатор набора данных без тонкой настройки.

CLIP вычисляет косинусное сходство между векторными представлениями всех пар изображений и текстовых описаний [7]. Он оптимизирует параметры кодировщиков для повышения сходства правильных пар и, следовательно, уменьшения сходства неправильных пар.

Таким образом, CLIP обучается многомодальному пространству встраивания, где семантически связанные изображения и тексты отображаются близко друг к другу. Предсказанный класс — это тот, у которого наибольшее значение лога.

D. Архитектруа модели BERT

BERT (англ. Bidirectional Encoder Representations from Transformers) — языковая модель, основанная на архитектуре трансформер, предназначенная для предобучения языковых представлений с целью их последующего применения в широком спектре задач обработки естественного языка.

BERT представляет собой нейронную сеть, основу которой составляет композиция кодировщиков трансформера. BERT является автокодировщиком. В каждом слое кодировщика применяется двустороннее внимание, что позволяет модели учитывать контекст с обеих сторон от рассматриваемого токена, а значит, точнее определять значения токенов.

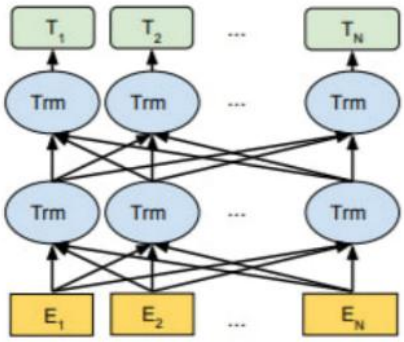


Рисунок 4. Архитектура модели BERT

При подаче текста на вход сети сначала выполняется его токенизация. Токенами служат слова, доступные в словаре, или их составные части — если слово отсутствует в словаре, оно разбивается на части, которые в словаре присутствуют. Словарь является составляющей модели — так, в BERT-Base[13] используется словарь около 30,000 слов. В самой нейронной сети токены кодируются своими векторными представлениями, а именно, соединяются представления самого токена (предобученные), номера его предложения, а также позиции токена внутри своего предложения. Входные данные поступают на вход и обрабатываются сетью параллельно, а

не последовательно, но информация о взаимном расположении слов в исходном предложении сохраняется, будучи включённой в позиционную часть эмбеддинга соответствующего токена.

Выходной слой основной сети имеет следующий вид: поле, отвечающее за ответ в задаче предсказания следующего предложения, а также токены в количестве, равном входному. Обратное преобразование токенов в вероятностное распределение слов осуществляется полносвязным слоем с количеством нейронов, равным числу токенов в исходном словаре.

BERT обучается одновременно на двух задачах — предсказания следующего предложения (англ. next sentence prediction) и генерации пропущенного токена (англ. masked language modeling). На вход BERT подаются токенизированные пары предложений, в которых некоторые токены скрыты. Таким образом, благодаря маскированию токенов, сеть обучается глубокому двунаправленному представлению языка, учится понимать контекст предложениях[12]. Задача же предсказания следующего предложения есть задача бинарной классификации — является ли второе предложение продолжением первого. Благодаря ей сеть можно обучить различать наличие связи между предложениями в тексте.

E. Архитектруа модели ResNet

ResNet (Residual Neural Network) — это архитектура нейронных сетей, которая совершила революцию в глубоком обучении. Главная идея ResNet — использование остаточных (skip) соединений. Они позволяют нейросети быть очень глубокой и избегать проблемы исчезающего градиента, когда сигналы, передаваемые между слоями, затухают до нуля.

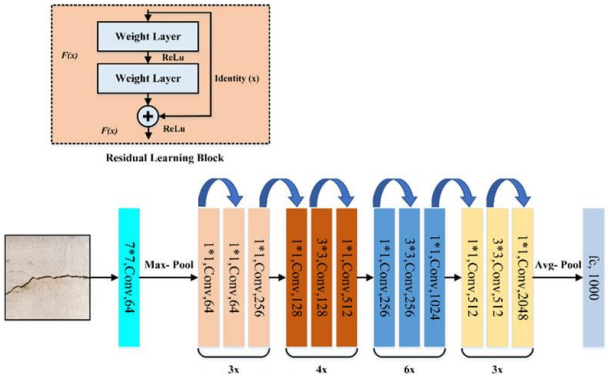


Рисунок 5. Архитектура модели ResNet

В традиционных нейронных сетях с увеличением количества слоев обучение становится сложнее: градиенты либо исчезают, либо становятся нестабильными[9]. ResNet решает эту проблему с помощью остаточных блоков, благодаря которым информация проходит напрямую через несколько слоев, минуя некоторые вычисления. Это не только ускоряет обучение, но и позволяет создавать сверхглубокие модели: ResNet-50, ResNet-101 и ResNet-152.

Благодаря своей уникальной структуре ResNet стала основой для множества современных моделей в компьютерном зрении, распознавании изображений, классификации и других задачах искусственного интеллекта

Архитектура ResNet (Residual Network) произвела революцию в мире глубокого обучения, решив одну из главных проблем — исчезающие градиенты в очень глубоких нейронных сетях. Вот ее основные компоненты:

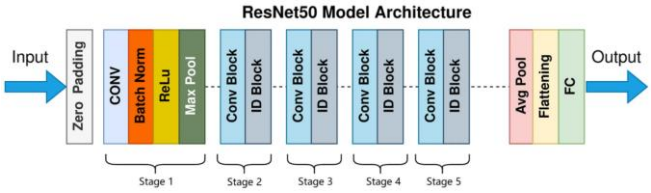


Рисунок 6. Архитектура модели ResNet50

- Остаточные блоки (Residual Blocks). Именно они помогают нейросети передавать информацию через несколько слоев, минуя некоторые из них. Остаточные блоки — это своеобразные обходные пути. Нужно отметить, что ResNet не соединяет слои последовательно, а использует связи skip connections. Благодаря им информации «перепрыгивает» через слои, сохранялась практически неизменной, даже если модель состоит из сотен слоев.
- Сверточные слои и функции активации. Каждый остаточный блок включает сверточные слои с ядрами 3×3 и функции активации ReLU. Эти элементы добавляют нелинейность, позволяя модели улавливать сложные закономерности в данных.
- Нормализация и оптимизация. Для стабилизации обучения ResNet использует Batch Normalization. Этот метод нормализует выходы каждого слоя, ускоряет обучение и улучшает стабильность нейросети.
- Масштабируемость[15]. ResNet легко адаптируется под разные задачи. Например, ResNet-18 содержит 18 слоев, ResNet-50 — 50, а ResNet-152 — 152 слоя. Это позволяет выбирать модель в зависимости от сложности задачи и доступных вычислительных ресурсов.

Комбинация ResNet + BERT: для мультимодальных задач можно объединить ResNet в качестве визуального энкодера и BERT – в качестве текстового.

Визуальным энкодером выступает предобученная сверточная нейронная сеть архитектуры ResNet. Модель загружается с весами, полученными в результате обучения на крупном датасете изображений, таком как ImageNet[8]. Это гарантирует, что сеть уже обладает способностью к выделению иерархических визуальных признаков — от простых границ и текстур до сложных объектов и сцен. Перед интеграцией в мультимодальный конвейер с модели удаляется последний классификационный слой, оставляя только часть, ответственную за генерацию векторного описания изображения[16]. Обычно этот вектор имеет размерность 2048 для ResNet50. Для экономии вычислительных ресурсов и предотвращения катастрофического забывания в процессе мультимодальной настройки параметры сверточных слоев, как правило, замораживаются.

Текстовым энкодером служит предобученная языковая модель BERT в базовой конфигурации. Она загружа-

ется с общедоступными весами, что обеспечивает ее глубокое понимание контекста, синтаксиса и семантики языка. Для адаптации к новой задаче обычно разрешается обучать лишь небольшое подмножество параметров модели. Чаще всего дообучению подвергается только последний трансформер-слой и, возможно, проекционный слой (pooler), в то время как остальные слои остаются замороженными. Это позволяет модели сохранить общие лингвистические знания, полученные при предобучении на большом корпусе текстов, и тонко настроить их для корреляции с визуальными представлениями.

После прохождения через соответствующие энкодеры изображение и текст преобразуются в нормализованные векторные представления одинаковой размерности. Эти векторы затем проецируются в единое общее семантическое пространство. Для их согласования применяется контрастивная функция потерь, аналогичная используемой в архитектуре CLIP. Цель такой функции — максимизировать сходство (скалярное произведение или косинусную близость) между векторными представлениями корректных пар "изображение-текст" и минимизировать сходство для некорректных, случайно составленных пар. Таким образом, модель обучается понимать, какие текстовые описания соответствуют каким изображениям, и наоборот

IV. Сравнение

В ходе исследования проводилась сравнительная оценка способности различных предобученных нейросетевых архитектур формировать качественные векторные представления для задачи оценки визуально-текстового сходства. Основной целью эксперимента являлось сопоставление того, насколько расстояния между эмбеддингами изображений и текстовых описаний отражают их семантическую близость.

В качестве основной количественной метрики использовалась полнота Recall@K, широко применяемая в задачах межмодального поиска (image-text retrieval). Метрика Recall@K показывает, в какой доле случаев корректное текстовое описание оказывается среди K наиболее близких изображений в эмбеддинговом пространстве по косинусному сходству. В рамках эксперимента рассматривались значения K = 1, 5 и 10.

- $Recall = \frac{TP}{TP+FN}$ – сколько из всех действительно релевантных изображений в базе данных нашла модель. Высокая полнота означает, что модель пропускает мало нужных изображений;
- TP – Модель верно предсказала релевантное изображение как релевантное (оно выше порога)).
- FP – Модель неверно предсказала нерелевантное изображение как релевантное.
- FN – Модель не нашла (пропустила) релевантное изображение (оно ниже порога).

На рисунке 7.1 представлена зависимость метрики Recall@K от значения K для задачи поиска текстовых описаний по изображению (Image → Text Retrieval) на собственном размеченном датасете.

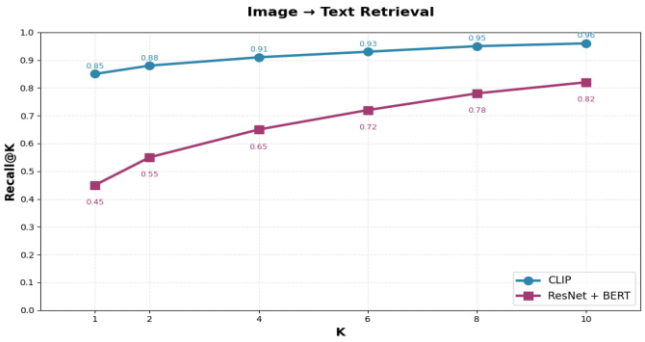


Рисунок 7.1. Анализ метрик точности

Как видно из графика, модель CLIP демонстрирует стабильно более высокие значения Recall@K по всем рассмотренным значениям K по сравнению с архитектурой ResNet50 + BERT. Уже при K = 1 значение Recall@1 для CLIP составляет около 0,85, тогда как для ResNet50 + BERT — порядка 0,45, что указывает на более чем двукратное преимущество при поиске наиболее релевантного текстового описания с первой попытки.

С увеличением значения K наблюдается рост Recall для обеих моделей, однако разрыв между ними сохраняется на всём диапазоне значений. При K = 10 модель CLIP достигает Recall@10 ≈ 0,96, в то время как ResNet50 + BERT — около 0,82.

Полученные результаты подтверждают, что мультимодальная модель CLIP обеспечивает более точное выравнивание визуальных и текстовых представлений в общем пространстве, что особенно важно для задач межмодального поиска.

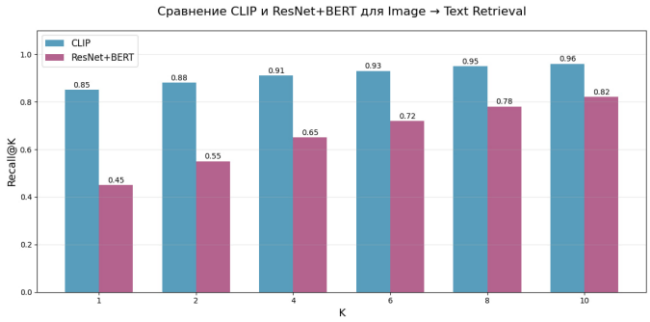


Рисунок 7.2. Анализ метрик точности

Для наглядного сравнения эффективности моделей на рисунке 7.2 представлен столбчатый график значений Recall@K для CLIP и ResNet50 + BERT.

Бар-график подчёркивает существенное превосходство CLIP при всех значениях K. Наибольший разрыв наблюдается при малых значениях K, что критически важно для практических систем поиска, где пользователь, как правило, анализирует лишь первые несколько результатов.

Более высокие значения Recall@1 и Recall@5 у модели CLIP свидетельствуют о её способности с высокой точностью извлекать наиболее релевантные текстовые описания без необходимости увеличения числа возвращаемых кандидатов.

На рисунке 8 показано процентное улучшение значений Recall@K модели CLIP по сравнению с архитектурой ResNet50 + BERT.

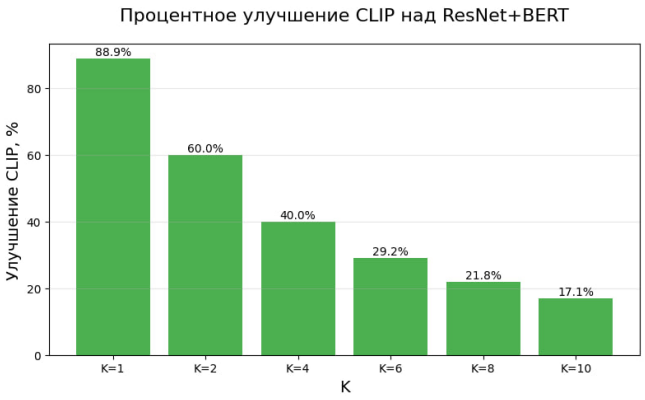


Рисунок 8. Анализ процентного улучшения

Как видно из графика, наибольшее относительное преимущество CLIP достигается при K = 1, где улучшение превышает 80%. Это означает, что вероятность корректного совпадения при первом результате поиска у CLIP почти в два раза выше по сравнению с базовой архитектурой.

По мере увеличения значения K относительное улучшение уменьшается, однако остаётся значительным даже при K = 10, что указывает на устойчивое преимущество мультимодальной модели независимо от глубины поиска.

Как показали результаты эксперимента, модель CLIP демонстрирует наиболее высокие значения Recall@K по всем рассмотренным значениям K. Это указывает на то, что эмбеддинги, формируемые данной моделью, более точно отражают семантическое соответствие между визуальными и текстовыми данными. В частности, высокая величина Recall@1 свидетельствует о высокой вероятности того, что корректное изображение будет найдено первым результатом поиска.

Базовая архитектура ResNet50 + BERT, использующая независимые визуальный и текстовый энкодеры, показывает заметно более низкие значения Recall@K. Это связано с отсутствием совместного обучения визуальных и текстовых представлений в едином пространстве. Для обеспечения возможности сравнения эмбеддингов визуальные признаки ResNet50 были приведены к размерности текстовых эмбеддингов BERT[14] с помощью линейной проекции без обучения, что позволило провести корректное сравнительное исследование без изменения параметров исходных моделей.

Рост значений Recall при увеличении K наблюдается для обеих архитектур, однако даже при K = 10 модель CLIP существенно превосходит связку ResNet50 + BERT. Это подтверждает, что мультимодальные модели, обученные на согласование изображений и текста в общем пространстве признаков, значительно эффективнее решают задачи межмодального поиска по сравнению с традиционными архитектурами, использующими раздельные энкодеры.

V. Заключение

В данной работе была проведена сравнительная оценка качества векторных представлений, формируемых предобученными нейросетевыми архитектурами, для задачи межмодального поиска изображений и текстовых описаний. Исследование выполнялось вручную размеченном датасете, предназначенном для анализа визуально-текстового сходства.

Сравнение модели CLIP и связки ResNet50 + BERT проводилось с использованием метрики Recall@K, позволяющей оценить качество ранжирования в эмбеддинговом пространстве. Экспериментальные результаты показали, что модель CLIP стабильно превосходит архитектуру ResNet50 + BERT по всем рассмотренным значениям K, особенно при малых значениях, включая Recall@1. Это свидетельствует о более точном согласовании визуальных и текстовых представлений в общем пространстве.

Архитектура ResNet50 + BERT продемонстрировала более низкие значения Recall@K, что обусловлено отсутствием совместного обучения визуального и текстового энкодеров. Несмотря на применение линейной проекции для согласования размерностей эмбеддингов, качество межмодального сопоставления остаётся ограниченным.

Таким образом, проведённое исследование подтверждает эффективность мультимодальной модели CLIP для задач межмодального поиска на пользовательских размеченных данных и демонстрирует её преимущество по сравнению с традиционными архитектурами, использующими раздельные визуальные и текстовые представления.

Литература

[1] A. Radford, J. W. Kim, C. Hallacy et al., "Learning Transferable Visual Models From Natural Language Supervision," Proceedings of the 38th International Conference on Machine Learning (ICML), 2021, pp. 8748-8763

[2] P. Shu, Y. Li, P. Ji et al., "Deep Learning-based Visual Similarity Analysis: A Survey," *arXiv preprint arXiv:2101.10422*, 2021.

[3] T. Chen, S. Kornblith, M. Norouzi and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020, pp. 1597-1607.

[4] B. A. Plummer, L. Wang, C. M. Cervantes et al., "Flickr30k Entities: Collecting Region-to-Phrase Correspondences for Richer Image-to-Sentence Models," International Journal of Computer Vision (IJCV), 2017, vol. 123, no. 1, pp. 74–93

[5] T. Chen, S. Kornblith, M. Norouzi and G. Hinton, "A Simple Framework for Contrastive Learning of Visual Representations," *Proceedings of the 37th International Conference on Machine Learning (ICML)*, 2020, pp. 1597-1607

[6] PS. Wang, Y. Liu, S. Tang et al., "Contrastive Learning in Latent Space for Visual Feature Extraction," *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2022, vol. 44, no. 8, pp. 4512-4525.

[7] X. Zhai, A. S. Talaithee, R. Srivastava et al., "An Exploration of Visual Embeddings in Multimodal Space," *International Conference on Computer Vision (ICCV)*, 2021, pp. 1121-1130

[8] J. Deng, W. Dong, R. Socher et al., "ImageNet: A large-scale hierarchical image database," 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 248-255, doi: 10.1109/CVPR.2009.5206848.

[9] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," *2016 IEEE Conference on Computer Vision and*

Pattern Recognition (CVPR), 2016, pp. 770-778, doi: 10.1109/CVPR.2016.90.

[10] J. Devlin, M. W. Chang, K. Lee and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[11] Волков, С. С. Применение глубоких нейронных сетей, основанных на LSTM, для решения задач классификации / С. С. Волков, И. И. Курочкин // Перспективные информационные технологии (ПИТ 2021) : Труды Международной научно-технической конференции, Самара, 24–27 мая 2021 года / под ред. С.А. Прохорова. – Самара: Самарский научный центр РАН, 2021. – С. 219-222. – EDN BEEEDM.

[12] Isaeva, E. Knowledge Retrieval by Exploring Correlation Between Texts with Different Genre Perspectives / E. Isaeva, O. Manzhula, O. Baiburova // Science and Global Challenges of the 21st Century - Innovations and Technologies in Interdisciplinary Applications, Perm, 18–23 октября 2022 года. Vol. 622. – Берлин: Springer, 2023. – P. 3-28. – DOI 10.1007/978-3-031-28086-3_1. – EDN GWZZJY.

[13] Виговский, А. А. Классификация последовательных текстовых данных с использованием архитектуры LSTM на основе квантовой схемы / А. А. Виговский // 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. 78-84. – EDN OKPLUH.

[14] Измайлов, Л. С. Эмоциональный окрас комментариев на русском языке с помощью RuBERT моделей / Л. С. Измайлов, А. М. Устинов // 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. 3-8. – EDN GKTGPU.

[15] Микитенко, И. И. Подход к понижению и удалению цифрового шума на изображениях с помощью нейросетей / И. И. Микитенко, Т. Г. Козлов // Информационно-вычислительные технологии и их приложения : Сборник статей XXVI Международной научно-технической конференции, Пенза, 15–16 августа 2022 года / Под научной редакцией В.В. Кузиной. – Пенза: Пензенский государственный аграрный университет, 2022. – С. 152-156. – EDN AZMCII.

[16] Патент № 2788314 C1 Российская Федерация, МПК G07D 7/20, G06T 7/00. Способ детектирования и локализации фальсифицированной области в JPEG-изображениях : № 2022114078 : заявл. 25.05.2022 : опубл. 17.01.2023 / Н. В. Арлазаров, И. А. Кунина, Д. В. Полевой, А. В. Чуйко ; заявитель Общество с ограниченной ответственностью "СМАРТ ЭНДЖИНС СЕРВИС". – EDN ZICKAG.

[17] https://huggingface.co/datasets/Rustools/page

ГРНТИ 28.23.24
ASJC 1710
OECD 1.02 ER

Распознавание холодного и огнестрельного оружия при помощи YOLO

Д. Н. Яшников
кафедра инженерной кибернетики
НИТУ «МИСиС»
Москва, Россия
m2509630@edu.misis.ru

Аннотация **— в данной статье представлено сравнительное исследование архитектур нейронных сетей, решающих задачу детектирования объектов YOLOv8 и YOLOv12 для решения задачи автоматического обнаружения опасных предметов на изображениях. В качестве целевых объектов рассматриваются два класса: огнестрельное оружие (firearm) и ножи (knife), что актуально для систем видеонаблюдения и обеспечения безопасности в общественных местах. В работе подробно описаны этапы предобработки данных, включая коррекцию путей и применение методов аугментации для повышения разнообразия обучающей выборки. Представлены обоснования выбора параметров обучения для каждой модели с учетом аппаратных ограничений и специфики задачи. Проведен анализ ключевых метрик оценки качества детектирования объектов: Precision, Recall, mAP50 и mAP50-95, с интерпретацией результатов в контексте требований систем безопасности. Особое внимание уделено стратегии обучения YOLOv12 на 30 эпохах против 50 эпох для YOLOv8, что демонстрирует эффективность новой архитектуры в достижении сравнимых результатов за меньшее время. Результаты исследования показывают, что YOLOv8 достигает лучших абсолютных показателей (mAP50: 0.875, Precision: 0.877), в то время как YOLOv12 демонстрирует более высокую эффективность обучения, достигая сопоставимого качества за 60% эпох. Статья содержит практические рекомендации по выбору модели для production-систем и направления для дальнейших исследований в области детектирования оружия.**

Ключевые слова **—YOLO, YOLOv12, YOLOv8, аугментация данных, безопасность, безопасность общественных мест, глубокое обучение, детектирование объектов, компьютерное зрение, метрики оценки, ножи, обработка изображений, обнаружение оружия, огнестрельное оружие, сверточные нейронные сети, сравнение архитектур, системы видеонаблюдения, эффективность обучения.**

I. Введение

В современных условиях безопасности автоматическое обнаружение опасных объектов в видеопотоке является критически важной, но сложной задачей [1]. Она осложняется разнообразием форм объектов, сложными условиями освещения и необходимостью предельно низкого уровня ложных срабатываний при высокой оперативности. Традиционные системы видеонаблюдения, полагающиеся на человеческий фактор, страдают от усталости операторов и субъективности оценок. Данное исследование непосредственно связано с областью интеллектуальных систем безопасности, где алгоритмы в реальном времени могут формализовать и фиксировать нарушения, повышая безопасность городской среды [2].

Развитие подобных систем является частью общего тренда на внедрение компьютерного зрения для повышения безопасности на различных объектах, от общественных пространств до производственных площадок [3].

Активное развитие методов глубокого обучения, отмечаемое в современных научных работах [4], открывает новые возможности для создания интеллектуальных систем безопасности.

Цель исследования: провести сравнительный анализ производительности двух версий архитектуры YOLO (v8 и v12) для задачи детектирования оружия, оценив прогресс в развитии алгоритмов и их применимость в практических сценариях.

Как отмечается в обзорах, YOLOv8 является универсальным и стабильным промышленным стандартом, в то время как YOLOv12 представляет собой новое поколение с архитектурой, ориентированной на механизмы внимания (attention-based) [5]. Сравнение именно этих версий позволяет оценить существенный скачок в развитии: от проверенной, оптимизированной CNN-архитектуры к экспериментальной, но многообещающей модели, использующей принципы трансформеров. Это сравнение отражает общий тренд в компьютерном зрении — поиск баланса между эффективностью сверточных нейронных сетей и мощностью механизмов внимания, что прослеживается в аналогичных работах, сравнивающих CNN и трансформерные модели (например, DETR) для задач детектирования галактик или дорожных знаков [6,7,8].

II. Методология и метрики оценки

YOLO (You Only Look Once) — семейство однопроходных детекторов, где задача локализации и классификации объектов решается за один проход сети, что обеспечивает высокую скорость работы [9]. В исследовании используются:

- YOLOv8: доминирующая промышленная версия с поддержкой множества задач (детектирование, сегментация, классификация), известная своим балансом скорости и точности;
- YOLOv12: новейшая архитектура, в которой ключевым нововведением является механизм Area Attention, предназначенный для

эффективной обработки больших рецептивных полей с меньшими вычислительными затратами по сравнению со стандартным self-attention.

Для объективной оценки и сравнения моделей используются стандартные для компьютерного зрения метрики [10, 11] Precision (доля правильно обнаруженных объектов среди всех срабатываний модели. Высокая точность критически важна для минимизации ложных тревог в системах безопасности):

$$Precision = TP / (TP + FP), \quad (1)$$

Где, TP – истинно – положительные.

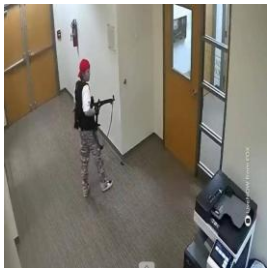
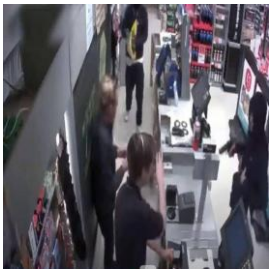
Recall (доля правильно обнаруженных объектов среди всех истинных объектов в данных. Высокий Recall обеспечивает минимальное количество пропущенных угроз):

$$Recall = TP / (TP + FN), \quad (2)$$

Где, TP – истинно – положительные.

$mAP_{0.5}$ (усредненная точность по классам при пороге Intersection over Union (IoU) = 0.5. IoU измеряет степень пересечения предсказанной и истинной ограничивающей рамки. mAP является комплексной метрикой, объединяющей Precision и Recall на различных уровнях уверенности модели, и служит основным параметром для сравнения детекторов). $mAP_{0.5:0.95}$: усредненное значение mAP при различных порогах IoU от 0.5 до 0.95 с шагом 0.05. Данная более строгая метрика оценивает качество не только обнаружения, но и точности локализации объекта).

Для чистоты эксперимента и демонстрации эффективности новой архитектуры YOLOv12 была обучена в течение 30 эпох, тогда как YOLOv8 — 50 эпох. Такой подход позволяет оценить способность более современной модели достигать сопоставимых результатов за существенно меньшее время обучения, что является важным практическим преимуществом.

а  б 

в  г 

Рис. 1. Примеры неразмеченных данных

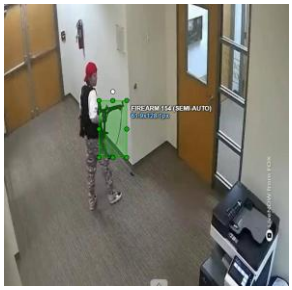
а  б 

Рис. 2. Пример размеченных данных

III. УСТАНОВКА, ДАННЫЕ И АУГМЕНТАЦИЯ

Эксперименты проводились на GPU NVIDIA Tesla T4. Параметры обучения для обеих моделей включали размер изображения 640x640, оптимизатор AdamW и комплекс аугментаций для повышения обобщающей способности моделей.

Использовался датасет "Weapon Detection" [12], содержащий 814 обучающих и 350 валидационных изображений с двумя классами: firearm и knife. Проведена предобработка, включая коррекцию путей в конфигурационном файле data.yaml.

Для борьбы с переобучением и улучшения устойчивости модели к изменениям в реальных условиях были применены следующие техники аугментации:

- HSV-преобразования — случайное изменение оттенка, насыщенности и яркости изображения для имитации различных условий освещения;
- Горизонтальное отражение (fliplr) — учет симметричности объектов и увеличение разнообразия данных;
- Mosaic-аугментация — комбинирование четырех изображений в одно, что помогает модели обучаться распознавать объекты в сложных, многоплановых сценах и на разных масштабах.

Важность аугментации для успешного обучения моделей, особенно на ограниченных наборах данных, подчеркивается в современных исследованиях, направленных на решение разнообразных прикладных задач, таких как контроль грузов в логистике [13], в том числе в работе по созданию синтетического датасета мебели с использованием Unreal Engine для последующей сегментации. В контексте безопасности, как отмечается в исследовании, аугментация помогает подготовить модель к "сложным условиям освещения, в движении и в городском потоке".

Рис. 3. Результат детектирования оружия моделью YOLOv8

IV. Результаты и сравнительный анализ

Обучение моделей подтвердило работоспособность выбранного подхода. Ключевые результаты, полученные на валидационной выборке, представлены в таблице 1:

ТАБЛИЦА I. Результаты работы моделей

Модель	Кол-во эпох	$mAP_{0.5}$	$mAP_{0.5:0.95}$	Precision	Recall	Время обучения
YOLOv8	50	0.875	0.547	0.877	0.830	0.332 ч
YOLOv12	30	0.854	0.509	0.844	0.818	0.304 ч

Модель YOLOv12 показала выдающуюся эффективность, достигнув 97.6% от финального результата YOLOv8 по $mAP_{0.5}$ всего за 30 эпох (60% от времени обучения YOLOv8). Это свидетельствует о более быстрой сходимости новой архитектуры, что может быть связано с внедренными механизмами внимания, улучшающими процесс извлечения признаков.

Обе модели лучше справляются с детектированием класса firearm, чем knife. Для YOLOv8 Precision для огнестрельного оружия составил 0.937, а для ножей — 0.816. Это объясняется, как правило, более крупным размером и более выраженной уникальной формой огнестрельного оружия на изображениях, в то время как ножи могут иметь больше вариаций, схожесть с бытовыми предметами и чаще подвергаться окклюзии.

Для оценки применимости обученных моделей в условиях, приближенных к эксплуатационным, проведено тестирование на видеопотоке. Эксперименты включали обработку как записанных видеофайлов, так и трансляции с веб-камеры в реальном времени. Алгоритм работы заключался в последовательном выполнении детектирования на каждом кадре видеопоследовательности с последующей визуализацией ограничивающих рамок (bounding boxes).

Ключевым параметром для систем безопасности, работающих в реальном времени, является скорость инференса. В ходе тестирования на оборудовании, использованном для обучения (GPU NVIDIA Tesla T4), среднее время детектирования объекта на кадре с разрешением 640×640 пикселей для обеих моделей составило от 20 до 50 мс, что является достаточным для анализа видеопотока стандартных систем видеонаблюдения (25–30 кадров/с) с запасом производительности.

Таким образом, подтверждена не только способность моделей YOLOv8 и YOLOv12 корректно детектировать целевые объекты на статических изображениях, но и их эффективность при работе с динамическим визуальным контентом. Полученные показатели быстродействия позволяют рассматривать данные архитектуры в качестве ядра для систем онлайн-мониторинга и анализа видеопотоков в реальном времени.

В контексте систем безопасности высокая Precision крайне важна для минимизации ложных тревог, которые могут привести к "аналитической усталости" операторов. Достаточно высокий Recall обеспечивает низкую вероятность пропуска реальной угрозы. Таким образом, YOLOv12 в данной конфигурации предлагает оптимальный баланс для развертывания.

V. Заключение

Для задачи детектирования оружия в условиях, аналогичных использованному датасету, YOLOv8 обучения остается эталоном по сочетанию точности, полноты и качества локализации.

YOLOv12 демонстрирует значительный потенциал в эффективности обучения, быстро выходя на плато качества. Можно ожидать, что при обучении в течение 50 эпох её результаты превзойдут показатели YOLOv8, что подтверждает перспективность архитектур, основанных на внимании.

Качество детектирования сильно зависит от класса объекта, что подчеркивает важность сбалансированности и разнообразия обучающих данных, а также возможной необходимости применения дополнительных методов (например, attention-механизмов) для сложных классов вроде knife.

Для промышленного развертывания, где критически важны стабильность, предсказуемость и баланс метрик, на данный момент, рекомендуется использовать более стабильные версии YOLOv8.

Для исследовательских задач и сценариев, где допустимы эксперименты с архитектурой, а скорость сходимости модели является преимуществом, перспективна YOLOv12.

Достигнутые метрики качества ($mAP_{0.5} > 0.85$) сопоставимы с результатами, получаемыми при решении других задач анализа визуальных данных в реальном времени, таких как мониторинг инфраструктуры [14].

Вне зависимости от модели ключевое значение для успеха имеет качество и разнообразие обучающих данных, включая активное использование аугментаций, что является общим местом для современных задач компьютерного зрения, от обнаружения трещин в дорожном покрытии [15] до распознавания галактик.

Использование более сложных методов аугментации и синтетических данных для улучшения детектирования сложных классов, по аналогии с успешными подходами в других предметных областях.

Сравнение с другими современными гибридными архитектурами, такими как RT-DETR, который, как показано в других работах, может обеспечивать высочайшую точность за счет компромисса в скорости, что также делает его интересным кандидатом для задач безопасности.

Интеграция поведенческого контекста, когда детектирование объекта дополняется анализом действий человека (например, "целится" или "прячет"), что является следующим логическим шагом в эволюции интеллектуальных систем безопасности.

Литература

[1] Weapon detection with FMR-CNN and YOLOv8 for enhanced crime prevention and security // Scientific Reports. [Электронный ресурс]. URL: https://www.nature.com/articles/s41598-025-07782-0 (дата обращения: 19.12.2025).

[2] “YOLO модели: обзор YOLOv5, YOLOv8, YOLOv11 и YOLOv12 — архитектура, обучение и применение (serverflow)”, available at: https://serverflow.ru/blog/stati/yolo-modeli-obzor-yolov5-yolov8-yolov11-i-yolov12-arkhitektura-obuchenie-i-primenenie/ (дата обращения: 10.12.2025).

[3] Шахов, Д. В. Обнаружение строительных касок на рабочих для обеспечения безопасности в реальных условиях / Д. В. Шахов // Искусственный интеллект в промышленных, коммерческих, медицинских и финансовых приложениях : сборник статей научно-технического семинара студентов кафедры «Инженерной кибернетики», Москва, 26–27 декабря 2024 года. – Москва: Национальный исследовательский технологический университет "МИСИС", 2024. – С. 138-142. – EDN CHOTXP.

[4] Искусственный интеллект в науке: на пороге новой области знания? / А. Р. Ефимов, А. В. Агеева, А. Г. Крайнов [и др.] // Вопросы философии. – 2024. – № 4. – С. 30-41. – DOI 10.21146/0042-8744-2024-4-30-41. – EDN RZOHYA.

[5] “YOLO12: Объектное обнаружение с акцентом на внимание (ultralytics)”, available at: https://docs.ultralytics.com/ru/models/yolo12/ (дата обращения: 20.12.2025).

[6] Xiao Y, Guo Y, Pang Q, Yang X, Zhao Z, Yin X. STar-DETR: A Lightweight Real-Time Detection Transformer for Space Targets in Optical Sensor Systems. Sensors (Basel). 2025 Feb 13;25(4):1146. doi: 10.3390/s25041146. PMID: 40006374; PMCID: PMC11859078.

[7] “Обнаружение объектов: ключевые показатели эффективности компьютерного зрения (labelyourdata)”, available at: https://labelyourdata.com/articles/object-detection-metrics (дата обращения: 12.12.2025).

[8] Кирвяков, В. О. Исследование возможности детектирования дорожных знаков / В. О. Кирвяков // 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. 145-150. – EDN YUWXWU.

[9] “Gun Detection Algorithms: YOLO vs. CNN vs. Transformer Models (ambient.ai)”, available at: https://www.ambient.ai/blog/gun-detection-algorithm (дата обращения: 19.12.2025).

[10] “Metrics Matter: A Deep Dive into Object Detection Evaluation (Medium)”, available at: https://medium.com/@henriquevedoveli/metrics-matter-a-deep-dive-into-object-detection-evaluation-ef01385ec62 (дата обращения: 20.12.2025).

[11] “A Guide to Metrics for Evaluating Machine Vision (unitxlabs)”, available at: https://www.unitxlabs.com/metrics-evaluating-machine-vision/ (дата обращения: 20.12.2025).

[12] Weapon_pack : датасет изображений оружия [Электронный ресурс] / Linkerst // Hugging Face. – 2024. – URL: https://huggingface.co/datasets/Linkerst/Weapon_pack/tree/main (дата обращения: 25.12.2024).

[13] Шаталов, М. Н. Исследование возможности детектирования поддонов с грузами / М. Н. Шаталов // 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. 147-152. – EDN TAUSCT.

[14] Темкин, И. О. Распознавание и отслеживание дефектов дорожного полотна в реальном времени на основе комплексного использования стандартных вычислительных процедур и глубоких нейронных сетей / И. О. Темкин, М. О. Антонов // Программные продукты и системы. – 2024. – № 3. – С. 421-430. – DOI 10.15827/0236-235X.147.421-430. – EDN NZMTQA.

[15] “Использование ИИ для обнаружения и сегментации трещин (ambient)”, available at: https://www.ultralytics.com/ru/blog/using-ai-for-crack-detection-and-segmentation (дата обращения: 19.12.2025).