Санкт-Петербургский государственный университет

Кафедра системного программирования

Группа 22.Б15-мм

Сравнение открытых движков OCR в ADAS на задаче распознавания дорожных знаков с навигационной информацией

Ахмедов Давид Хусенович

Отчёт по учебной практике
в форме «Сравнение»

Научный руководитель:
старший преподаватель кафедры системного программирования, к.т.н., Литвинов Ю. В.

Консультант:
Инженер-программист АО «Кама», Осечкина М. С.

Санкт-Петербург
2023

Оглавление

Введение

Управление дорожно-транспортным средством представляет собой сложную задачу, требующую постоянного внимания к дорожной обстановке, контроля за множеством параметров автомобиля, принятия быстрых решений в различных ситуациях и учета множества факторов, влияющих на безопасность движения. В связи с этим, на дорогах всегда существует опасность дорожно-транспортных происшествий из-за человеческих ошибок.

Для повышения безопасности на дорогах и облегчения управления транспортным средством в последние годы широко используются системы ADAS (Advanced Driver Assistance Systems). Эти системы предоставляют различные функции поддержки водителя, такие как контроль полосы движения, контроль слепых зон, автоматическое торможение и другие. Путем предупреждения об опасности и активного вмешательства в процесс управления автомобилем, системы ADAS помогают водителям уменьшить вероятность возникновения аварийных ситуаций. Так они играют важную роль в повышении безопасности на дорогах и обеспечении более комфортного управления автомобилем.

OCR (Optical Character Recognition) на дороге используется для распознавания и чтения текстовой информации, такой как дорожные знаки, указатели направления, номерные знаки автомобилей и т.д. Некоторые из этих задач уже решены: например, такие средства используются на статических камерах для отслеживания правонарушений по автомобильным номерам. Для локализации на дороге используется GPS, однако точности этой системы может не хватать даже для правильного проезда сложной развязки. Путем распознавания текста на указателях направления можно улучшить точность локализации на дороге и обеспечить удобство для водителя на развязках.

В рамках данного исследования планируется выбор и последующее сравнение открытых движков OCR в контексте ADAS. Основное внимание будет уделено оценке эффективности данных систем в условиях реальной эксплуатации на дорожных указателях.

1. Постановка задачи

Целью работы является актуальных движков для распознавания текста в ADAS на задаче распознавания текста дорожных знаков. Для её выполнения были поставлены следующие задачи:

1. Исследовать и выбрать инструменты, наиболее подходящие под специфику задачи.
2. Разработать метод для валидации распознаваемого с изображения текста.
3. Разработать метод для оценки расстояния до дорожных знаков.
4. Измерить качество распознавания текста движками в реальных условиях и численно сравнить их.

2. Обзор

2.1. Известные движки

1. Tesseract — на данный момент самый известный OCR-движок, т.к. является одним из самых первых серьезных решений для распознавания символов. Tesseract является одним из самых известных и широко используемых движков для распознавания текста. Хотя настройка и оптимизация Tesseract может быть сложной задачей, это универсальный инструмент, не специализированный для определенной задачи, и может быть использован в ADAS.

2. EasyOCR является более легковесным конкурентом Tesseract. Для извлечения текста на изображениях EasyOCR обучает многослойные нейронные сети. Движок может распознавать текст на различных языках и имеет широкий набор поддерживаемых языков.

3. Keras OCR — OCR-движок общего назначения. Как и EasyOCR, использует многослойные нейронные сети для обучения моделей.

4. GOCR не подходит для систем реального времени, где критично время распознавания [8].

5. ocropus — ответвление Tesseract, специализированное для документов.

А также kraken — ответвление ocropus, оптимизированное для исторических материалов и материалов, написанных нелатинским шрифтом и ocrad — OCR-движок для документов.

2.2. Существующие работы

На данный момент существует недостаточно исследований, охватывающих все аспекты и особенности OCR-движков. Некоторые области, требующие распознавания текста в реальном мире, как ADAS, практически не исследованы.

«Comparative Analysis of EasyOCR and TesseractOCR for Automatic License Plate Recognition using Deep Learning Algorithm [9]». Исследование, в котором рассмотрены EasyOCR и Tesseract на задаче распознавания автомобильных номеров. Для сравнения использовалась метрика Character Error Rate (CER)[7]. Результаты показали, что EasyOCR справляется лучше Tesseract при распознавании цифр, и наоборот с буквами. Итоговая точность распознавания оказалась 95% с EasyOCR и 88% с Tesseract.

«Advanced Driver Assistance Systems (ADAS) Based on Machine Learning Techniques for the Detection and Transcription of Variable Message Signs on Roads [3]». В работе был предложен метод для получения текста с электронных дорожных знаков с переменной информацией (VMS), используя Tesseract. Подобран способ предобработки изображения, который состоит из нескольких этапов и предположительно может улучшить результаты распознавания:

1. Поворот распознанного на изображении объекта (выпрямление).
2. Перевод изображения в двухцветное черно-белое (бинаризация).
3. Устранение разрывов в контурах объектов с помощью закрытого морфологического преобразования.

Однако текст может распознаваться некорректно (больше всего от этого страдают изображения низкого качества), т.к. символы на изображении могут стать неразборчивыми даже для человека ввиду того, что контуры на изображении могут стать значительно шире.

«Comparison of OCR [1]». В работе предоставлен открытый инструмент для сравнения результатов OCR на изображении с использованием Keras OCR, Tesseract, EasyOCR. Некоторые выводы автора:

1. Для улучшения результатов с Tesseract морфологическое преобразование действительно полезно. Также Tesseract менее эффективно распознает текст с изображений, на которых несколько областей с текстом. То есть используя этот OCR-движок, рекомендуется предварительно локализовать каждую текстовую область.

2. Keras OCR показал высокую точность, но он не подходит для систем реального времени, т.к. требует больших вычислительных ресурсов.

3. EasyOCR легковесный и хорошо справляется при работе со структурированным текстом, а также с зашумленными изображениями.

Таким образом, Tesseract и EasyOCR представляют собой наиболее предпочтительные варианты выбора, поскольку они являются движками общего назначения с высокой производительностью и отличаются хорошей точностью распознавания. Ввиду ограниченных возможностей, для начала были выбраны Tesseract и EasyOCR, хотя стоит отметить, что OCR-движки, специализированные для документов, тоже нужно исследовать и сравнивать.

Для улучшения результатов распознавания текста с помощью Tesseract может быть полезно применение морфологического преобразования. EasyOCR может быть более точным при работе со структурированным текстом и зашумленными изображениями.

3. Метод

3.1. Извлечение текста с дорожного знака

Для того, чтобы извлечь текст с дорожного знака по кадру, изображение нужно отфильтровать, избавившись от стороннего текста, не обладающей навигационной информацией. Например, это могут быть автомобильные номера, рекламные баннеры или вывески.

Рис. 1: Пример обнаружения текста без фильтрации

Одним из возможных методов фильтрации может быть предварительная локализация дорожных знаков на кадре, что позволит не только избавиться от лишнего текста, но и распознавать текст с дорожных знаков по отдельности. Преимуществом этого метода является простота, однако существенным недостатком может быть скорость распознавания объектов, т.к. это тяжелая процедура.

3.1.1. Одноэтапные и двухэтапные модели распознавания объектов

Одноэтапные модели распознавания объектов выполняют обнаружение и классификацию объектов в один проход и могут быть более пригодными для использования в системе реального времени, т.к. работают быстрее, но в то же время могут иметь более низкую точность по сравнению с двухэтапными моделями.

Двухэтапные модели распознавания объектов: на первом этапе модель обнаруживает регионы интереса (ROI), которые могут содержать объекты, далее эти регионы интереса подвергаются классификации. Двухэтапные модели могут обеспечивать более высокую точность, так как они могут более тщательно анализировать регионы интереса, но они могут быть более трудоемкими в обучении и требовать больше вычислительных ресурсов.

3.1.2. Выбор модели

Благодаря исследованию актуальных моделей на задаче распознавания автомобилей[2] среди самых известных одноэтапных моделей — YOLOv8[12], RetinaNet[4] и Single Shot Detector[5] — был получен вывод о том, что YOLOv8 является наиболее предпочтительным выбором среди самых известных одноэтапных моделей: она демонстрирует наилучшую производительность по времени распознавания, точности и наиболее предпочтительна для обеспечения возможности использования в режиме реального времени. Этим обусловлен выбор YOLOv8.

3.2. Сбор эталонных данных и обучение модели

Для обучения модели под специфичную задачу нужен большой датасет: большой объем данных позволит модели обучиться на разнообразных условиях освещения, погодных условиях, углах наклона объекта и других переменных, что сделает ее более надежной и эффективной в реальных условиях эксплуатации.

К сожалению, датасетов с размеченными дорожными знаками с навигационной информацией нет в свободном доступе, поэтому нужно самостоятельно размечать изображения. Для сбора эталонных данных использовался отрезок видеозаписи с ездой на автомобиле по Москве.

3.2.1. Извлечение кадров с OpenCV

OpenCV — библиотека для компьютерного зрения, поддерживающая работу на Python и C++. Для извлечения кадров из видеопотока с

последующей разметкой объектов был разработан инструмент с использованием этой библиотеки. В целях оптимизации производительности был выбран язык C++, так как OpenCV показывает более высокую эффективность на этом языке по сравнению с Python.

Так с помощью инструмента было извлечено 450 кадров с дорожными знаками. Для повышения точности прогнозов целесообразно расширить объем эталонных данных, однако процесс разметки является трудоемким. В случае необходимости увеличения точности распознавания, возможно дополнительное расширение датасета.

3.2.2. Разметка дорожных знаков и создание датасета

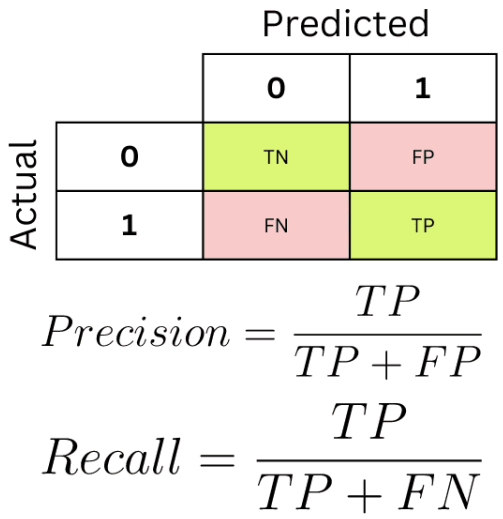
Для процесса разметки объектов был задействован Roboflow[1] – инновационный онлайн-инструмент, специально разработанный для удобной и эффективной разметки эталонных данных. Этот инструмент обладает уникальной способностью предоставления результатов в разнообразных форматах, включая поддержку YOLOv8, что позволяет с легкостью интегрировать полученные данные в различные среды и проекты. С помощью данного сервиса можно увеличить разнообразие эталонных данных, не размечая изображения дополнительно: размеченные на изображениях объекты возможно автоматически преобразить с помощью инвертирования, сдвига и поворота. Так датасет был увеличен до 923 изображений.

3.2.3. Обучение модели

Метрики. Precision — отношение числа верно распознанных объектов к общему числу гипотез о распознанных объектах, или же $\frac{TP}{TP+FP}$. Recall — доля обнаруженных объектов, или же $\frac{TP}{TP+FN}$. mAP_{50}[11] (mean average precision) — средняя точность, рассчитанная при пересечении с пороговым значением IoU = 0.5 (мера пересечения с эталонными данными). mAP_{50-95} же дает представление о точности модели на

[1] https://roboflow.com/ (дата доступа: 19 декабря 2023 г.).

различных уровнях уверенности в обнаружении, учитывая mAP для классов с IoU от 0.5 до 0.95 с шагом 0.05.



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

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

Рис. 2: Иллюстрация метрик Precision и Recall
Источник:https://www.kdnuggets.com/2022/11/confusion-matrix-precision-recall-explained.html

Обучение. Для обучения выбрана модель YOLOv8m в виду того, что данная модель предлагает точность по метрике mAP_{50}, сопоставимую с более точными YOLOv8l и YOLOv8x (разница в точности — 5-6%), в то же время затрачивая значительно меньше времени на детекцию — в 1.5-2 раза.

Обучение прошло всего 200 поколений, т.к. модель с продолжением обучения начала показывать регресс и ее оценка mAP_{50-95} стала уменьшаться. Пример распознавания дорожных знаков можно посмотреть на видео в Google Drive[2].

Гиперпараметры

model	YOLOv8m
epochs	200
image resolution	640

Результаты обучения модели

Precision	0.99722
Recall	0.97059
mAP_{50}	0.98973
mAP_{50-95}	0.98262

[2]https://drive.google.com/drive/folders/1Xvj-KU7tA2wFod_hddeINJpd1hZ1er4O?usp=sharing

3.3. Препроцессинг

Зная специфику предметной области, с помощью предобработки мы можем помочь движку в распознавании, улучшив его точность и дав ему сосредоточиться на ключевых деталях изображения.

3.3.1. Выпрямление изображения

С помощью преобразования Хафа[6] можно найти линии на изображении. О каждой найденной линии нам известны координаты в полярной системе координат – пара (r, θ), где $r \in \mathbb{Q}$ — расстояние до прямой, $\theta \in [0, \pi)$ – угол наклона прямой к оси ординат. Положим, что знак не может быть под наклоном более, чем $\frac{\pi}{4}$, чтобы разделить прямые на два класса: прямая горизонтальная, если $\theta \in [0, \frac{\pi}{4}) \cup [\frac{3\pi}{4}, \pi)$, иначе – вертикальная. Теперь приведем все θ к отрезку $[-\frac{\pi}{4}; \frac{\pi}{4}]$ с помощью f:

$$f(\theta) = \begin{cases} \theta, & \text{if } \theta \in [0, \frac{\pi}{4}) \\ \theta - \pi/2, & \text{if } \theta \in [\frac{\pi}{4}, \frac{3\pi}{4}) \\ \theta - \pi, & \text{if } \theta \in [\frac{3\pi}{4}, \pi) \end{cases}$$

Если повернуть изображение по какой-то прямой на $f(\theta)$, то она станет перпендикулярна оси абсцисс, если горизонтальная, и перпендикулярна оси ординат, если вертикальная. Для нахождения угла поворота для списка из $f(\theta_1),\ f(\theta_2), ..., f(\theta_n)$ найдем все значения, которые отклоняются от среднего не более, чем на два стандартных отклонения, чтобы хотя бы частично избавиться от всплесков в данных, не относящихся к контуру дорожного знака. Также будет полезно ввести пороговое значение для угла поворота изображения, чтобы уменьшить вероятность «испортить» изображение из-за всплесков — не более 0.05 (rad) по модулю.

3.3.2. Бинаризация

Для бинаризации используем метод Otsu[3] для автоматического нахождения пороговых значений.

3.3.3. Морфологическое преобразование

Применим закрытое морфологическое преобразование для устранения «пробелов» в контурах символов. Стоит отметить, что такая процедура эффективна на изображениях с высоким качеством, но из-за потери деталей она способна ухудшать результаты на изображениях низкого качества.

3.4. Оценка расстояния до дорожного знака

Для точной оценки расстояния необходимо знать скорость движения и о параметрах каждого дорожного знака, но получить эти данные сложно. Из соображений оптики в предположении, что горизонт на кадре параллелен сторонам, можно найти приближение расстояния до объекта.

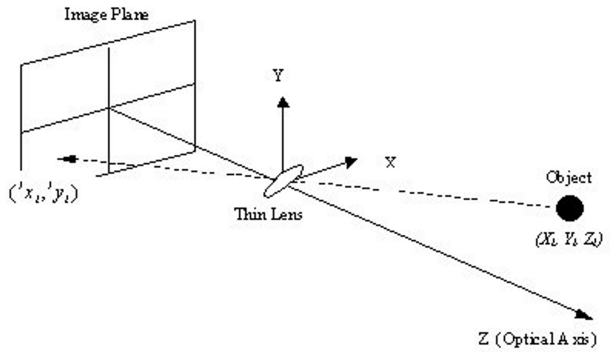


Рис. 3: точка, расположенная за линзой, проецируется на плоскость изображения, но координаты будут зеркально отражены в обоих направлениях

Источник: https://www.researchgate.net/publication/239744055_3D_Vision-Based_Control_On_An_Industrial_Robot

[3] https://docs.opencv.org/3.4/d7/d4d/tutorial_py_thresholding.html (дата доступа: 19 декабря 2023 г.).

Пусть f — фокальное расстояние объектива камеры (в пикселях), hor_s — горизонт сцены на кадре, hor_o — горизонталь объекта на кадре, pp — принципиальная точка, h — разница между высотой объекта и высотой подвеса камеры. Рассмотрим векторы $\vec{a} = (0, hor_s - pp.y, f)$, $\vec{b} = (1, 0, 0)$ и векторное произведение $a \times b$, образующее уравнение плоскости дороги: $f * y + (pp.y - hor_s) * z + c = 0$. Подставим $(0, h, 0)$ и найдем c: $c = -f * h$.

Уравнение плоскости дороги, смещенной на h по Oy:

$$y = h_{cam} + \frac{hor_s - pp.y}{f} * z$$

С векторами $\vec{a_1} = (0, pp.y - hor_o, f)$ и $b_1 = (1, 0, 0)$ так же получим уравнение плоскости прообраза прямой $y = hor_o$ на камере (коэффициент обнулен):

Уравнение плоскости горизонтали объекта:

$$f * y + (pp.y - hor_o) * z = 0$$

С помощью пересечения плоскостей найдем расстояние до объекта:

$$z = \frac{f * h}{hor_o - hor_s}$$

По ГОСТу высота дорожного знака должна быть от 5 до 6 метров, для нашего эксперимента возьмем усредненное значение — 5.5. Горизонт определяется по видео и фиксирован для него, за горизонталь объекта возьмем крайнюю нижнюю точку распознанного дорожного знака.

3.5. Выбор стабильного кадра

Сначала из видео необходимо извлечь текст с дорожных знаков на видео. Чтобы определить, распознался ли дорожный знак, введем два пороговых значения: минимальный объем распознанных слов – w и минимальный объем распознанных символов – c (w, $c \in [0, 1]$). Для сравнения строк между собой используем расстояние Левенштейна[10] –

$Lev : str * str \to int$. Список, состоящий из списков строк, состоящих из слов дорожного знака — GTD (Ground Truth Data). Распознанные строки — DS (Detected Strings).

Сформулируем критерий стабильного распознавания текста: положим, что было распознано $D = [d_1, d_2, .., d_k] \in$ DS. Если существуют элемент $G = [g_1, g_2, ..., g_n] \in GTD$, по крайней мере p элементов из D (обозначим их за $T = [t_1, t_2, ..., t_p]$, где $t_i \in D$), $p \geq w * |GTD|$ и можно построить инъективное отображение из T в G, что для любого $t_i \in T$ найдется $g_j \in G$: $Lev(t_i, g_j) \leq c|g_j|$, то текст распознается верно.

3.6. Постпроцессинг

Для того, чтобы помочь движку в распознавании слов, можно исключить из результата распознавания те символы, которых нет в исходном словаре. Этим мы сможем уменьшить расстояние Левенштейна и увеличить вероятность правильно распознать слово.

4. Эксперимент

4.1. Условия эксперимента

Для экспериментов использовалось видео с ездой по МКАД[4]. Данное видео интересно тем, что известны параметры модели камеры, а также на видео разные погодные условия. Сравним качество распознавания с препроцессингом и без него. За пороговые значения w, c возьмем 0.5 и 0.5 соответственно. Отрезок с ясной погодой содержит 59 дорожных знаков, с дождливой – 139. Предварительно видео было обрезано по вертикали с 1080 пикселей до 900, чтобы улучшить производительность, избавившись от капота автомобиля. Итоговое разрешение видео — 1920*900 пикселей.

Испытания проводились с процессором Intel Xeon 2.20 GHz[5] и 30GB RAM, используя Kaggle, сервисом, свободно предоставляющим вычислительные ресурсы.

4.2. Исследовательские вопросы

- Какое среднее время распознавания?
- На каком расстоянии от машины текст начинает распознаваться?
- Насколько качество распознавания зависит от погодных условий?
- Оценка ошибок первого и второго рода.
- Может ли препроцессинг помочь в распознавании?

4.3. Результаты

Эксперимент показал, что Tesseract распознает текст почти в два раза быстрее. EasyOCR удалось распознать больше дорожных знаков,

[4] https://youtu.be/j_CAOYYs4d0 МКАД за 15 минут. Видеорегистратор Xiaomi 70mai Dash Cam. (дата доступа: 16 декабря 2023 г.).

[5] https://www.kaggle.com/code/teeyee314/cpu-kernel-specs Kaggle's CPU Kernel Specs (дата доступа: 16 декабря 2023 г.).

однако он имеет более высокую среднюю сумму расстояний Левенштейна, что говорит о том, что при интерпретации результатов распознания может быть сложно восстановить исходное слово. Оба движка способны распознать текст на расстоянии 8-9 метров (для выбранных пороговых значениях).

В дождливую погоду EasyOCR справился заметно лучше и практически не потерял в точности, в отличие от Tesseract: для EasyOCR разница в доле распознанных объектов оказалась 1%, и 6% для Tesseract. Возможно, EasyOCR менее чувствителен к помехам на изображении.

Оценка ошибок первого рода на дорожный знак приведена в таблицах 1, 2 («Среднее число неверно распознанных слов»). Оценкой ошибок второго рода будет разница между средним количеством верно распознанных слов и средним количеством распознанных слов («Среднее число верно распознанных слов» и «Среднее число распознанных слов» соответственно).

Таблица 1: Ясная погода (без препроцессинга)

	EasyOCR	Tesseract
Доля распознанных дорожных знаков	0.525	0.423
Среднее число слов	7.9	7.8
Среднее расстояние распознавания (м)	9.16	9.69
Среднее время распознавания (мс)	740.49	368.74
Среднее расстояние Левенштейна	7.65	5.76
Среднее число неверно распознанных слов	3.06	3.36
Среднее число верно распознанных слов	4.84	4.44
Среднее число распознанных слов	8.55	8.32

Таблица 2: Дождливая погода (без препроцессинга)

	EasyOCR	Tesseract
Доля распознанных дорожных знаков	0.517	0.366
Среднее число слов	8.31	8.35
Среднее расстояние распознавания (м)	8.44	9.53
Среднее время распознавания (мс)	790.56	476.02
Среднее расстояние Левенштейна	7.22	6.20
Среднее число неверно распознанных слов	3.64	3.65
Среднее число верно распознанных слов	4.67	4.71
Среднее число распознанных слов	8.57	9.86

Также заметно, что препроцессинг оказался негативным фактором в ясную погоду для Tesseract и в дождливую для EasyOCR: движки распознали меньшее количество дорожных знаков и лишь увеличили время распознавания. Хотя этим среднее растояние Левештейна уменьшилось, в остальном преимущество не смогло перешагнуть пределов погрешности.

Таблица 3: Ясная погода (с препроцессингом)

	EasyOCR	Tesseract
Доля распознанных дорожных знаков	0.491	0.355
Среднее число слов	7.93	7.76
Среднее расстояние распознавания (м)	9.14	9.79
Среднее время распознавания (мс)	793.16	448.25
Среднее расстояние Левенштейна	6.83	5.38
Среднее число неверно распознанных слов	3.28	3.52
Среднее число верно распознанных слов	4.66	4.24
Среднее число распознанных слов	8.83	8.19

Таблица 4: Дождливая погода (с препроцессингом)

	EasyOCR	Tesseract
Доля распознанных дорожных знаков	0.374	0.374
Среднее число слов	7.98	7.87
Среднее расстояние распознавания (м)	8.6	9.2
Среднее время распознавания (мс)	862.05	469.03
Среднее расстояние Левенштейна	7.10	5.71
Среднее число неверно распознанных слов	3.48	3.67
Среднее число верно распознанных слов	4.50	4.19
Среднее число распознанных слов	8.52	9.52

Среднее время распознавания дорожных знаков с YOLOv8 в ясную погоду оказалось 460мс со стандартным отклонением в 5мс, в дождливую — 465мс и 5мс соответсвенно.

4.4. Обсуждение результатов

Результаты показали, что EasyOCR хоть и требует больших вычислительных ресурсов для извлечения текста, однако может предоставить значительно большую точность распознавания текста, чем Tesseract, предоставляя хорошие результаты даже при плохих погодных условиях. Вероятно, более высокая точность распознавания текста EasyOCR может быть обусловлена применением более продвинутых алгоритмов и глубокого обучения нейронных сетей, что позволяет работать эффективнее с изображениями низкого качества. Хоть и эксперимент показал, что препроцессинг не оказался эффективным методом "помощи" движкам, на видео без сжатия предложенные преобразования могут улучшить результаты распознавания.

В данной работе тестировались модели, предоставляемые движками по умолчанию, однако с помощью их обучения на своих данных можно на порядок улучшить точность распознавания. Также использование GPU может значительно уменьшить время распознавания.

5. Применение

Применение результатов исследования может включать в себя интеграцию выбранного движка OCR в системы ADAS, что позволит автомобилям более точно распознавать текстовую информацию на дорожных объектах. Это, в свою очередь, способствует более точной локализации на дороге, предупреждению о возможных опасностях и обеспечению более надежной поддержки водителя. Результаты исследования имеют практическое значение для разработчиков автомобильных технологий, специалистов по безопасности дорожного движения и всех заинтересованных сторон, стремящихся улучшить безопасность и комфортность управления автомобилем.

Заключение

В ходе данной работы были выполнены следующие задачи:

- Разработан инструмент для анализа OCR по видео.
- Проведен анализ открытых OCR-движков, применимых в ADAS.
- Обучена модель распознавания дорожных знаков.
- Исследованы методы по улучшению результатов распознавания текста.

Реализация инструмента для анализа видео доступна по ссылке: https://github.com/Parzival-05/Comparison-of-OCR-ADAS.

Список литературы

[1] Akarsu Meftun. Comparison of OCR. — URL: https://github.com/mftnakrsu/Comparison-of-OCR (дата обращения: 5 декабря 2023 г.).

[2] Buitrón Iván Andrés. Performance Analysis of You Only Look Once, RetinaNet, and Single Shot Detector Applied to Vehicle Detection and Counting. — URL: https://link.springer.com/chapter/10.1007/978-3-031-45438-7_17 (дата обращения: 9 декабря 2023 г.).

[3] De-Las-Heras Gonzalo. Advanced Driver Assistance Systems (ADAS) Based on Machine Learning Techniques for the Detection and Transcription of Variable Message Signs on Roads. — URL: https://www.mdpi.com/1424-8220/21/17/5866 (дата обращения: 5 декабря 2023 г.).

[4] (FAIR) Facebook AI Research. Focal Loss for Dense Object Detection. — URL: https://arxiv.org/pdf/1708.02002.pdf (дата обращения: 19 декабря 2023 г.).

[5] Liu Wei. SSD: Single Shot MultiBox Detector. — URL: https://link.springer.com/chapter/10.1007/978-3-319-46448-0_2 (дата обращения: 19 декабря 2023 г.).

[6] OpenCV. Hough Transform. — URL: https://docs.opencv.org/3.4/d9/db0/tutorial_hough_lines.html (дата обращения: 14 декабря 2023 г.).

[7] Read-Coop. Character Error Rate. — URL: https://readcoop.eu/glossary/character-error-rate-cer/ (дата обращения: 19 декабря 2023 г.).

[8] Selinger Peter. Review of Linux OCR software. — URL: https://mathstat.dal.ca/~selinger/ocr-test/ (дата обращения: 7 декабря 2023 г.).

[9] Vedhaviyassh D.R. Comparative Analysis of EasyOCR and TesseractOCR for Automatic License Plate Recognition using Deep Learning Algorithm. — URL: https://ieeexplore.ieee.org/abstract/document/10009215 (дата обращения: 5 декабря 2023 г.).

[10] cuelogic. The Levenshtein Algorithm. — URL: https://www.cuelogic.com/blog/the-levenshtein-algorithm (дата обращения: 14 декабря 2023 г.).

[11] ultralytics. Performance Metrics Deep Dive. — URL: https://docs.ultralytics.com/guides/yolo-performance-metrics/#class-wise-metrics (дата обращения: 19 декабря 2023 г.).

[12] ultralytics. YOLOv8. — URL: https://github.com/ultralytics/ultralytics (дата обращения: 19 декабря 2023 г.).