СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире автомобильный транспорт играет важную роль в жизни людей. Однако, безопасность на дороге остается одной из главных проблем, и для решения этой проблемы необходимо обеспечить автомобилистов актуальной информацией о дорожной обстановке, в том числе о действующих на данном участке дороги знаках. В связи с этим, системы распознавания дорожных знаков являются важной областью в сфере автомобильной безопасности.

Эта технология может помочь водителям соблюдать правила дорожного движения и предотвратить возможные аварии. Также, автоматическое детектирование дорожных знаков может значительно упростить работу транспортных компаний и государственных органов, связанных с обслуживанием дорог.

Главной сложностью этой области является нетривиальность задачи распознавания. Условия внешней среды, такие как освещение, местоположение, окружающие объекты, постоянно меняются, вследствие чего усложняется и сама задача, поэтому на данный момент не существует канонического решения.

В данной работе будет рассмотрено несколько методов использования компьютерного зрения для автоматического детектирования дорожных знаков на изображениях и видео.

Целью работы является реализация алгоритмов детектирования, обучение модели сверточной нейронной сети для решения задачи классификации знаков, разработка программных модулей системы распознавания и сравнение разных подходов решения задачи.

Для достижения поставленной цели в работе необходимо решить следующие задачи:

1. Сформулировать постановку задачи.
2. Поиск и описание набора данных для исследования. Проведение предварительной обработки данных.
3. Реализация алгоритмов локализации знаков
4. Обучение модели сверточной НС для классификации дорожных знаков

5. Изучение и реализация алгоритма борьбы с отсечением неинформативных областей, полученных на этапе локализации.
6. Разработка программных модулей системы распознавания дорожных знаков
7. Оценка качества работы системы распознавания знаков. Анализ полученных результатов.

Глава 1. ПОСТАНОВКА ЗАДАЧИ РАСПОЗНАВАНИЯ ЗНАКОВ ДОРОЖНОГО ДВИЖЕНИЯ

1.1 Система распознавания дорожных знаков

Задача распознавания образов (изображений) – это применение алгоритмов классификации к данным, представленным в виде изображений. Распознавания дорожных знаков является одной из задач распознавания образов. Системы распознавания знаков дорожного движения, в основном, состоят из следующих компонентов (рисунок 1):

- Камера – устройство, которое фиксирует изображение знака на дороге.
- Процессор – устройство, которое обрабатывает полученную информацию и определяет, какой знак был зафиксирован.
- Алгоритмы детектирования – программное обеспечение, которое позволяет определить область знака в кадре.
- Алгоритмы распознавания –позволяют определить тип знака в кадре на основе изображения.
- База данных знаков дорожного движения – хранилище информации о всех возможных знаках, которые могут быть обнаружены на дороге.
- Интерфейс пользователя – часть системы, которая позволяет пользователю получать информацию о распознанных знаках, например, через экран на приборной панели автомобиля.

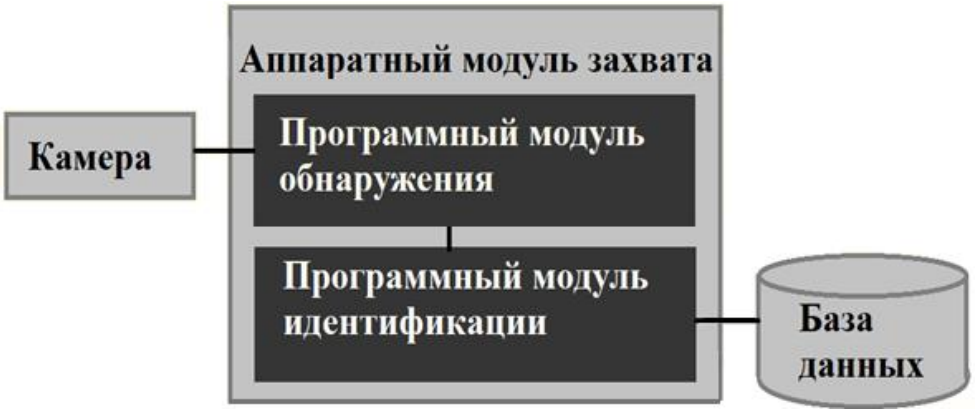

Рис.1 – Общая архитектура систем

С помощью камеры, изображение поступает на вход системы. Далее какой-либо алгоритм определяет положение знака дорожного движения. Далее дорожный знак распознается программным модулем идентификации.

Распознавание знаков можно разделить на две подзадачи:

1. Детектирование

   В этой подзадаче целью работы является найти область на изображении или видеопотоке, где есть дорожный знак. После чего вырезать изображение знака и подать в классификатор.

2. Классификация.

   В этой подзадаче целью работы является классифицировать изображения дорожных знаков и вывести результат о том, к какому классу принадлежит данный знак.

1.2 Задача детектирования.

Детектирование — это процесс обнаружения объектов на изображении или видео.

Одной из важных задач детектирования является обнаружение дорожных знаков на изображении или видео (рисунок 2). Особенность этой задачи заключается в том, что необходимо обнаружить объекты определенного класса (в данном случае - дорожные знаки) на изображении или видео сцене, которые могут иметь различные размеры, ориентации, положения и освещение. При этом необходимо учитывать возможные искажения, шумы и прочие помехи, которые могут повлиять на точность детектирования [1-2].

Рис.2 –Пример детектирования

Для решения этой задачи используются различные методы обработки изображений и машинного обучения, которые позволяют автоматически

находить дорожные знаки на изображениях и видео сценах с высокой точностью. Это может быть выполнено с помощью различных методов, таких как каскады Хаара, нейронные сети, методы, основанные на признаках объектов и т.д.

Детектирование дорожных знаков может быть полезным для автоматического распознавания знаков и предупреждения водителя об опасности на дороге, а также полезно для автоматических систем управления транспортом, таких как системы помощи водителю и автопилоты.

1.3 Задача классификации с помощью сверточной нейронной сети

Задача классификации заключается в определении класса, к которому относится объект на основе его признаков.

Для решения этой задачи используются алгоритмы машинного обучения, которые обучаются на базе данных объектов с известными классами. Эти алгоритмы позволяют автоматически классифицировать новые объекты, f результаты классификации могут быть использованы в различных областях, например, в медицине для диагностики заболеваний, в банковской сфере для определения кредитоспособности клиентов и т.д.

Математическая постановка задачи: пусть X – множество описаний объектов, Y – множество номеров наименований классов. Существует неизвестная целевая зависимость – отображение y*: X –> Y, значений которой известны только на объектах конечной обучающей выборки $X^m=\{(x_1,y_1),\ldots,(x_m,y_m)\}$. Требуется построить алгоритм a: X –> Y, способный классифицировать произвольный объект x, принадлежащий X.

В машинном обучении задача классификации может решаться с помощью применения искусственных нейронных сетей. Такой способ классификации представляет собой обучение с учителем.

Задача классификации изображений – это, непосредственно, применение алгоритмов классификации к данным в виде изображений.

Для решения этой задачи используются алгоритмы глубокого обучения, такие как сверточные нейронные сети. Эти алгоритмы обучаются на большой базе данных изображений с известными категориями и могут автоматически

классифицировать новые изображения с высокой точностью.

Задача классификации в контексте детектирования дорожных знаков заключается в определении типа и вариации знака на изображении.

Сверточные нейронные сети (Convolutional Neural Networks, CNN) [3] – это класс нейронных сетей, который широко используется для обработки изображений и видео. Они были разработаны в конце 1980-х годов и стали одним из наиболее популярных методов в области компьютерного зрения.

Основная идея сверточных нейронных сетей заключается в том, чтобы использовать операцию свертки для извлечения признаков из изображений. Операция свертки – это математическая операция, которая позволяет вычислить сумму произведений значений двух функций на каждом шаге. В контексте сверточных нейронных сетей, первая функция – это изображение, а вторая – это фильтр (ядро), который перемещается по изображению и вычисляет результаты свертки.

Фильтры (ядра) используются для извлечения различных признаков из изображений. Например, фильтры могут выделять границы объектов, текстуры, цвета и другие важные характеристики путем «скольжения» ядра по изображению, формируя карты признаков. При обучении сверточной нейронной сети, фильтры настраиваются автоматически, чтобы максимизировать точность распознавания.

Сверточные нейронные сети состоят из нескольких слоев. Первый слой – это входной слой, который принимает изображение в виде тензора формы N*H*W*C, где N – количество изображений, H – высота изображения, W – ширина изображения, C – количество каналов (1-если изображение в градациях серого и 3- если цветное). Затем следует несколько сверточных слоев, которые извлекают признаки из изображения. После этого идут слои объединения (pooling), которые уменьшают размерность признаков и улучшают обобщающую способность сети. Затем идут полносвязные слои, которые объединяют признаки и принимают решения о классификации изображения.

На рисунке 3 представлен наглядный принцип работы СНС, задачей которой было распознавание знаков дорожного движения.

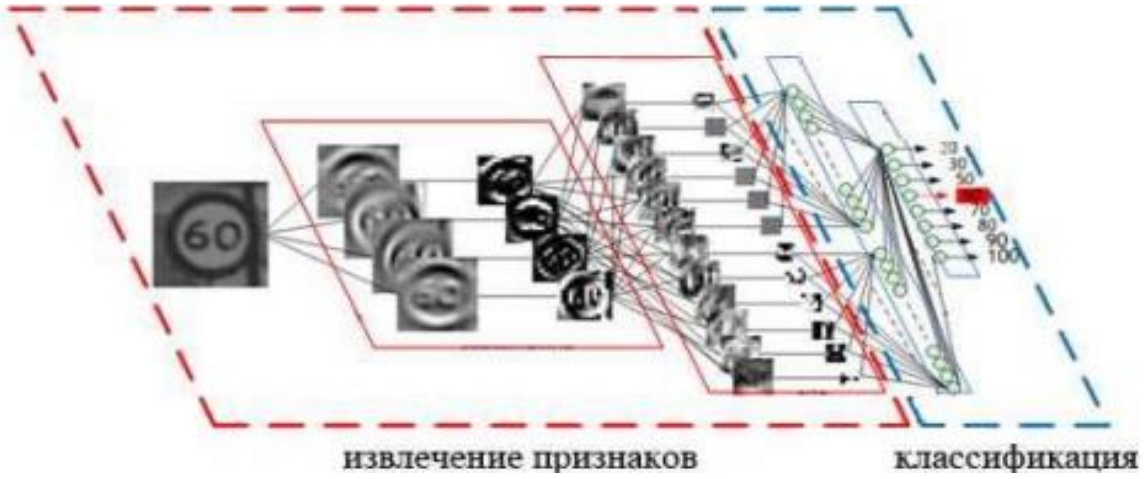

Рис 3 – Распознавание знака дорожного движения при помощи сверточной нейронной сети

Одна из главных причин, почему сверточные нейронные сети так популярны в обработке изображений, заключается в том, что они способны автоматически извлекать признаки из изображений без необходимости ручного определения характеристик. Кроме того, сверточные нейронные сети могут обрабатывать большие объемы данных и достигать высокой точности в распознавании объектов на изображениях.

Преимущества сверточных нейронных сетей:

- по сравнению с полносвязной сетью (типа перцептрон) – гораздо меньше количество настраиваемых обучаемых параметров
- инвариантность. То есть устойчивость к повороту изображения и сдвигу объекта на изображении

Наиболее простым и популярным способом обучения является метод обучения с учителем - метод обратного распространения ошибки.

1.4 Метрики качества

Матрица неточностей (Confusion matrix) является инструментом для оценки качества алгоритма машинного обучения в задачах классификации и детектирования. Она позволяет визуализировать, насколько успешно алгоритм классифицирует объекты каждого класса.

Матрица неточностей (ошибок) состоит из четырех значений: true positives (TP), false positives (FP), true negatives (TN) и false negatives (FN).

- True positive (TP) — количество верно классифицированных положительных примеров / количество верно детектированных знаков.
- True negative (TN) — количество верно классифицированных отрицательных примеров.
- False positive (FP) — количество неверно классифицированных положительных примеров / количество не найденных знаков в кадре.
- False negative (FN) — количество неверно классифицированных отрицательных примеров / количество неверно найденных знаков в кадре.

Матрица неточностей представлена в таблице 1:

Таблица 1 - Матрица неточностей

	Оценка эксперта	
Оценка классификатора	Положительная	Отрицательная
Положительная	TP	FP
Отрицательная	FN	TN

Используемые метрики в задаче классификации – аккуратность, точность, полнота, F-мера.

Аккуратность (англ. Accuracy) – доля правильных ответов алгоритма:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1.1)$$

Точность (англ. Precision) можно интерпретировать как долю объектов, названных классификатором положительными и при этом действительно являющимися положительными.

$$Precision = \frac{TP}{TP + FP} \quad (1.2)$$

Полнота (англ. Recall) показывает, какую долю объектов положительного класса из всех объектов положительного класса нашел алгоритм.

$$Recall = \frac{TP}{TP + FN} \tag{1.3}$$

F-мера – среднее гармоническое полнотой и точностью:

$$F = \frac{2 \times precision \times recall}{precision + recall} \tag{1.4}$$

Успешной классификацией знаков можно считать при значении метрик > 90%. Так же все описанные метрики можно применять и для оценки качества детектирования. Для этого используется следующая матрица неточностей, по которой рассчитываются все остальные метрики:
Матрица неточностей для детектирования представлена в таблице 2:

Таблица 2 - Матрица неточностей для детектирования

	Оценка эксперта	
Оценка классификатора	Объект есть в кадре	Объект отсутствует в кадре
Объект найден в кадре	TP	FP
Объект не найден в кадре	FN	TN

1.5 Выводы по главе

В первом разделе главы была рассмотрена общая постановка задачи распознавания дорожных знаков, составляющие системы и основные подзадачи.

Во втором и третьем разделах главы были подробнее рассмотрены подзадачи системы распознавания.

В четвертом разделе были описаны метрики оценки качества системы компьютерного зрения.

Глава 2. МЕТОДЫ И АЛГОРИТМЫ ДЕТЕКТИРОВАНИЯ ЗНАКОВ ДОРОЖНОГО ДВИЖЕНИЯ

В данной главе будут исследованы этапы детектирования (локализации) дорожных знаков. Основные методы решения данной задачи обычно подразделяются на:

1. Методы, опирающиеся на цветовые признаки знаков.
2. Методы, опирающиеся на форму знака.
3. Методы, основанные на машинном обучении (МО).

В данной работе будет рассмотрено 2 алгоритма детектирования:

1. Первый алгоритм будет опираться на информацию о форме знака. А для удаления шумов (задний фон знака (густой лес, дома), а также небо, которое создает слишком резкие перепады яркости), которые мешают алгоритму точнее определять форму объектов будет частично применен метод, опирающийся на цветовые признаки объектов.
2. Второй алгоритм будет реализован с помощью алгоритмов машинного обучения, и в качестве алгоритма, основанного на МО был выбран метод Виолы-Джонса (построение каскада классификаторов) из-за его высоких показателей точности [4], а так же наличие реализации данного метода в библиотеке OpenCV. [5]

2.1 Детектор границ Кэнни

Методы детектирования, основанные на форме дорожных знаков, являются одним из наиболее точных способов обнаружения знаков на изображениях. Они позволяют выделить объекты, имеющие определенную форму и размер, что делает их идеальным инструментом для поиска дорожных знаков.

Для начала необходимо определить форму и размер дорожных знаков. Для этого проводятся исследования форм и размеров различных знаков, которые позволяют определить характерные признаки каждого знака. Например, знак

"Остановка" имеет круглую форму, а знак "Осторожно дети"-треугольную.

После определения формы и размера знаков происходит выделение объектов на изображении, которые соответствуют этим признакам. Для этого используются алгоритмы обработки изображений, которые позволяют определить форму и размер объекта.

Далее происходит анализ выделенных объектов. Если объект соответствует форме и размеру дорожного знака, то он считается дорожным знаком. Для этого используются алгоритмы обработки изображений, которые позволяют определить форму и размер объекта.

Одним из наиболее популярных и точных алгоритмов поиска границ является «Детектор границ Кэнни». Он был разработан Джоном Кэнни в 1986 году и широко используется в области компьютерного зрения и обработки изображений. [6]

Задачей Кэнни была разработка оптимального алгоритма детектирования границ, который бы удовлетворял следующим требованиям:

- Высокий уровень обнаружение границ;
- Локализация, удовлетворяющая всем условиям (точное нахождение положение границы);
- На границу только один отклик.

Метод границ Кэнни основан на алгоритме, который позволяет выделить границы объектов на изображении. Он работает следующим образом:

1. Сглаживание изображения. Изображение сглаживается с помощью фильтра Гаусса, чтобы уменьшить шум и убрать мелкие детали.
2. Расчет градиента изображения. Градиент показывает направление и интенсивность изменения яркости на изображении. Он рассчитывается для каждого пикселя на основе сглаженного изображения.
3. Подавление не-максимумов. Для каждого пикселя на изображении определяется направление градиента, после чего проверяется, является ли

данный пиксель максимумом вдоль этого направления. Если нет, то он подавляется (рисунок 4).

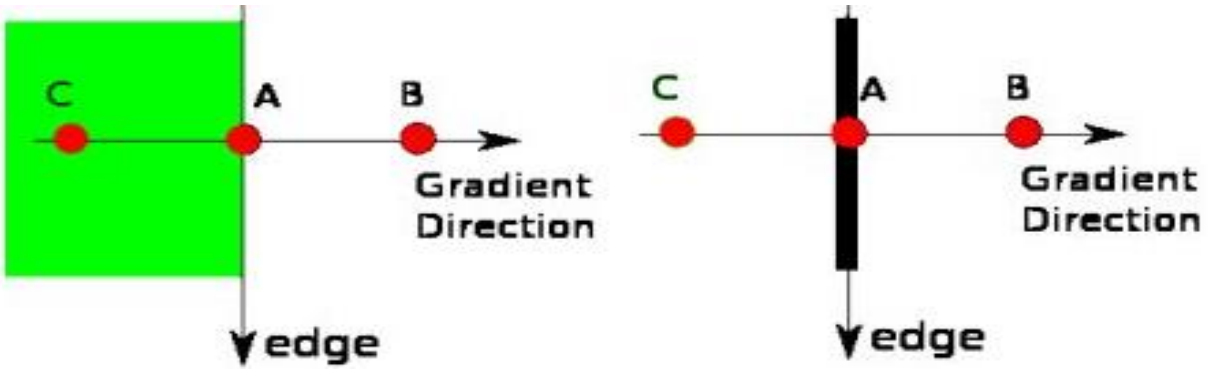

Рис.4 – подавление не-максимумов

Точка A находится на краю (в вертикальном направлении). Направление градиента нормально к краю. Точки B и C находятся в направлениях градиента. Таким образом, точка A сверяется с точками B и C, чтобы увидеть, образует ли она локальный максимум. Если это так, он рассматривается для следующего этапа, в противном случае он подавляется (обнуляется). Таким образом, в результате получается бинарное изображение с «тонкими краями». [7]

4. Двойная пороговая обработка. Пиксели, которые имеют интенсивность выше верхнего порога, считаются границами. Пиксели, которые имеют интенсивность ниже нижнего порога, отбрасываются. Пиксели, которые имеют интенсивность между двумя порогами, сохраняются только в том случае, если они связаны с пикселями, которые имеют интенсивность выше верхнего порога (рисунок 5).

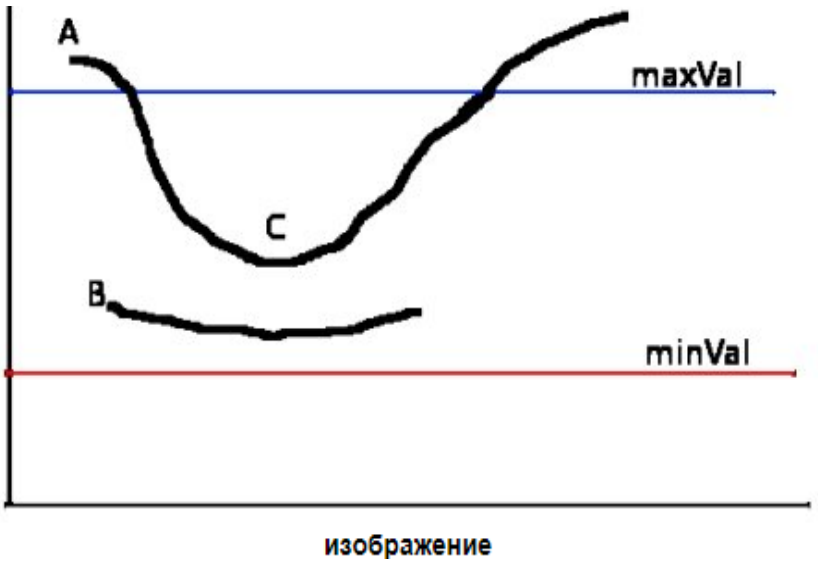

Рис.5 – двойная пороговая обработка

5. Тонкая обработка границ. Границы уточняются с помощью оператора Собеля, который позволяет определить точное местоположение границы.

Детектор границ Кэнни имеет ряд преимуществ перед другими методами обработки изображений. Он позволяет выделить только границы объектов на изображении, что делает его идеальным инструментом для поиска дорожных знаков. Кроме того, он позволяет получить точные результаты и имеет высокую скорость обработки изображений.

2.2 Детектор по цветовым признакам. Пространство HSV

Одним из самых распространенных методов обнаружения дорожных знаков является метод, основанный на цветовых признаках знаков. Этот метод использует информацию о цвете знака, чтобы выделить его на изображении. [8]

Для начала необходимо определить цветовые признаки знаков. Для этого проводятся исследования цветовых характеристик различных дорожных знаков, например, знак "Прямое направление" - синий, а знак "Ограничение максимальной скорости" - белый с красной окантовкой.

После определения цветовых признаков знаков происходит выделение объектов на изображении, которые соответствуют этим признакам. Для этого используются алгоритмы цветовой сегментации, которые позволяют выделить области изображения, имеющие определенный цвет.

Самым распространенным алгоритм цветовой сегментации является алгоритм, использующий пространство HSV (от англ. hue, saturation, value).

Изначально изображение представляется в пространстве RGB (от англ. red, green, blue), работа с таким форматом сильно затруднительна, т.к. обработка изображения в формате RGB довольно энергозатратная, а также изменение одной из составляющих, в цветовом пространстве, влияет на значения других составляющих, то есть, изменение G влияет на значения R и B.

Что бы избежать данных проблем, изображение представляют в формате HSV.

В формате HSV цвет представлен тремя параметрами:

- Оттенок (hue). Например, желтый, красный или синий.
- Насыщенность (saturation). Чем больше этот параметр, тем «чище» цвет, поэтому. А чем ближе этот параметр к нулю, тем ближе цвет к нейтральному серому.
- Значение (value)

Преимущества формата HSV перед RGB:

1. Удобство работы с цветом. В формате HSV цвет можно описать более естественно и интуитивно понятно. Например, можно легко изменить насыщенность или яркость цвета без изменения его оттенка.
2. Легкость выделения объектов по цвету. В формате HSV можно легко определить диапазон цветов, которые соответствуют определенному объекту на изображении. Это особенно полезно при обработке изображений в компьютерном зрении.
3. Совместимость с другими цветовыми моделями. Формат HSV может быть легко преобразован в другие цветовые модели, такие как RGB или CMYK.
4. Большая точность при работе с яркостью. В формате HSV яркость представлена отдельным параметром, что позволяет более точно работать с яркостью цвета

5. Меньшая чувствительность к изменениям освещения. Формат HSV менее чувствителен к изменению освещения, чем формат RGB, что позволяет получать более стабильные результаты при обработке изображений.

Таким образом, для изображения подбираются пороговые значения параметров H, S, V, которые соответствуют необходимому диапазону цветов.

После подстановки пикселя в промежуток сравниваются условия и если они соблюдаются, то пиксель становится белого цвета, если же нет, то черного. После того, как завершится задача проверки абсолютно всех пикселей, изображение станет черно-белым и, в частности, будут выделены только контуры знаков дорожного движения.

Одним из преимуществ метода, основанного на цветовых признаках знаков, является его простота и быстрота. Однако также этот метод является самым неустойчивым к внешним изменениям. Например, он может давать ложные срабатывания, если на изображении есть объекты, имеющие похожий цвет с дорожными знаками, засвеченные места, плохие условия освещения и т.д.

2.3 Метод Виолы-Джонса

Метод Виолы-Джонса — алгоритм компьютерного зрения, который используется для обнаружения объектов на изображении. Он основан на комбинации двух методов: метода интегральных изображений и каскадных классификаторов. [9]

Метод интегральных изображений представляет изображение в виде интегрального изображения, которое позволяет быстро вычислять суммы значений пикселей в любом прямоугольнике на изображении. Этот метод является одним из ключевых элементов метода Виолы-Джонса и широко используется в других алгоритмах компьютерного зрения.

Интегральное изображение создается путем преобразования каждого пикселя на исходном изображении в сумму значений всех пикселей, расположенных выше и левее данного пикселя.

Формула расчета значения элемента интегрального изображения:

$$S(x,y) = I(x,y) + S(x-1,y) + S(x,y-1) - S(x-1,y-1), \quad (2.1)$$

где S – результат предыдущих итераций для данной позиции пикселя,

I – яркость пикселя исходного изображения. Если координаты выходят за пределы изображения, они считаются нулевыми.

Пусть ABCD – интересующая нас прямоугольная область (рисунок 6).

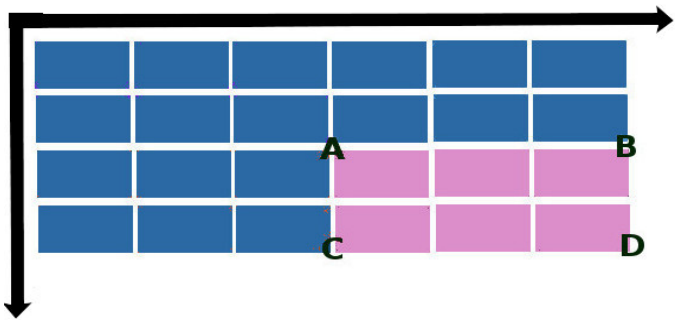

Рис.6 – Матрица изображения

Тогда суммарная яркость S в этой области вычисляется по формуле:

$$S(\text{ABCD}) = S(A) + S(D) - S(B) - S(C), \quad (2.2)$$

где S(A), S(B), S(C) и S(D) (рисунок 7):

Рис.7 – Значения элементов интегральной матрицы S(A) — S(D)

Это позволяет быстро вычислять суммы значений пикселей в любом прямоугольнике на изображении, используя только четыре значения интегрального изображения.

Метод каскадных классификаторов используется для обучения классификатора на основе набора признаков, которые были вычислены на интегральном изображении. Классификатор использует эти признаки для определения, является ли объект на изображении искомым объектом или нет.

Каждый этап каскадного классификатора состоит из нескольких слабых классификаторов, которые в результате представляют собой один сильный классификатор, который используются для проверки наличия признаков объекта на изображении. Если объект проходит проверку на одном этапе, он передается на следующий этап для более точной проверки. Если объект не проходит проверку на каком-то этапе, он сразу же отбрасывается, что позволяет значительно ускорить процесс обнаружения объектов на изображении. Пример алгоритма представлен на рисунке 8.

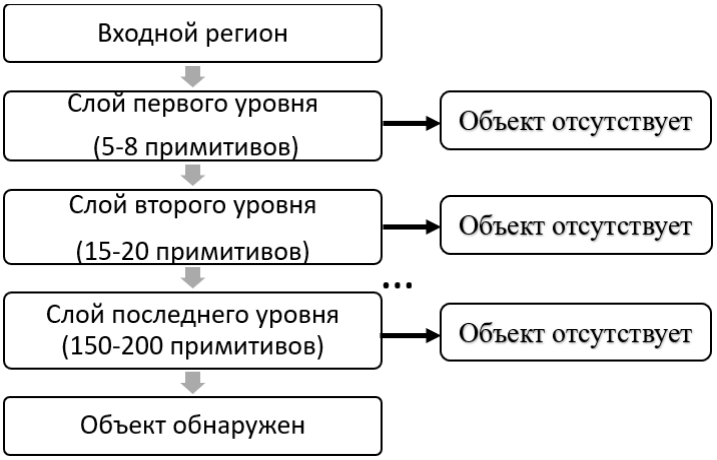

Рис.8 – Пример каскада классификаторов

В качестве слабых классификаторов используются признаки Хаара. Признаки Хаара представляют собой прямоугольные области на изображении, которые могут быть светлыми или темными. Эти области могут быть различной формы и размера, и могут быть расположены в разных частях изображения (рисунок 9).

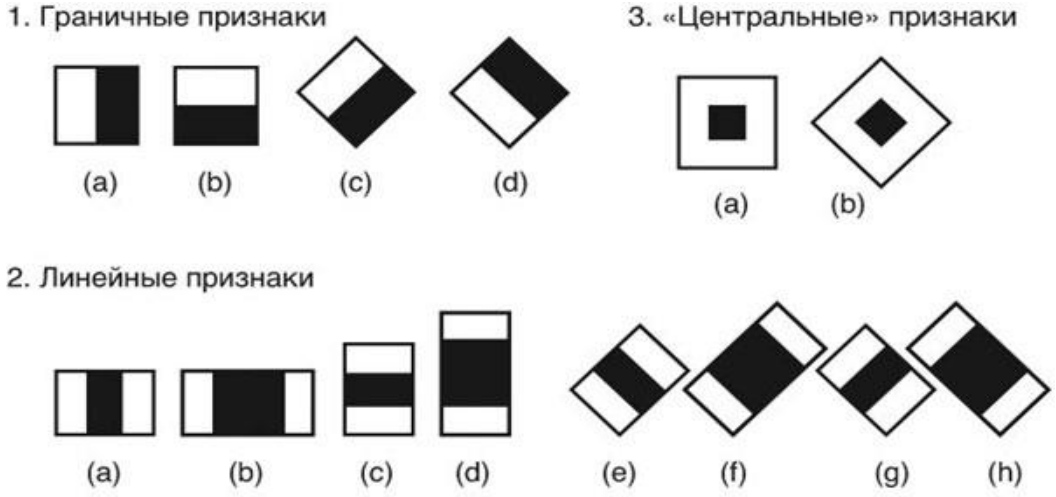

Рис.9– Признаки Хаара

Каждый признак Хаара представляет собой разность сумм пикселей в двух прямоугольных областях на изображении. Эта разность используется для определения наличия определенного признака объекта на изображении.

Признак Хаара чаще всего используется для обнаружения границы лица на изображении, однако он хорошо применим и к другим объектам распознавания, в том числе и знакам, которые имеют намного более тривиальную форму. Признак состоит из двух прямоугольных областей, один из которых находится на области объекта, а другой - за его пределами. Разность сумм пикселей в этих двух областях может указывать на наличие границы объекта.

Обучение каскадов происходит с помощью двух выборок: положительной и отрицательной. Положительная выборка состоит из изображений, содержащих дорожные знаки. Отрицательная не содержит знаков.

Само детектирование производится методом скользящего окна по всему изображению. По каждому окну вычисляется решение каскада и определяется, есть ли объект внутри текущего окна. Таким образом алгоритм анализирует изображение несколько раз: после каждого прохода по изображению, размер окна увеличивается (обычно на 20%) до определенного заданного значения. Чем медленнее будет увеличиваться окно, тем качественнее будет работать алгоритм, однако время обработки изображения так же вырастет.

Таким образом, Метод Виолы-Джонса можно считать одним из наиболее эффективных методов обнаружения объектов на изображении.

2.4 Выводы по главе

В первом разделе главы был изучен метод детектирования, опирающийся на формы объектов.

Во втором разделе был описан метод, опирающийся на цветовые признаки объектов

В третьем разделе главы был изучен алгоритм компьютерного зрения, основанный на алгоритмах машинного обучения.

Глава 3. РЕАЛИЗАЦИЯ АЛГОРИТМОВ ДЕТЕКТИРОВАНИЯ

В данной главе будут разработаны и протестированы два алгоритма детектирования дорожных знаков:

1. Первый алгоритм будет опираться на основе поиска контуров.
2. Второй алгоритм будет реализован с применением алгоритмов машинного обучения, а именно, построение каскада классификаторов (метод Виолы-Джонса).

Для проверки качества алгоритмов будут рассчитаны основные метрики качества, а именно: аккуратность, точность, полнота и F-мера.

Разрабатываться программы будут на языке Python с использованием библиотеки OpenCV.

OpenCV — библиотека компьютерного зрения с открытым исходным кодом, которая предоставляет различные алгоритмы для обработки изображений и видео, включая распознавание объектов, сегментацию изображений, детектирование лиц и многое другое.

3.1 Описание набора данных для решения задач детектирования

Для тестирования и обучения систем будет использован набор данных GTSDB (German Traffic Sign Detection Benchmark). Этот набор данных содержит изображения реальных сцен [10].

Он включает 900 изображений, из которых треть предназначена для проверки обнаружения дорожных знаков и 600 изображений для обучения этих систем. Все изображения цветные.

В обучающей выборке помимо изображений содержится файл «gt.txt» с информацией о местоположении знака в кадре, т.е. его координаты. Координаты представляют собой начальное положение (x, y) и изменение для каждого направления x и y (w, h). В сумме имеем набор: (x, y, x + w, y + h), представляющий собой обрамляющую рамку вокруг знака

Пример выборки немецких дорожных сцен представлен на рисунке 10.

Рис.10- Пример изображений из выборки

3.2 Алгоритм детектирования на основе метода границ Кэнни

Для разработки данного алгоритма представим его в виде Блок-схем (рисунок 11).

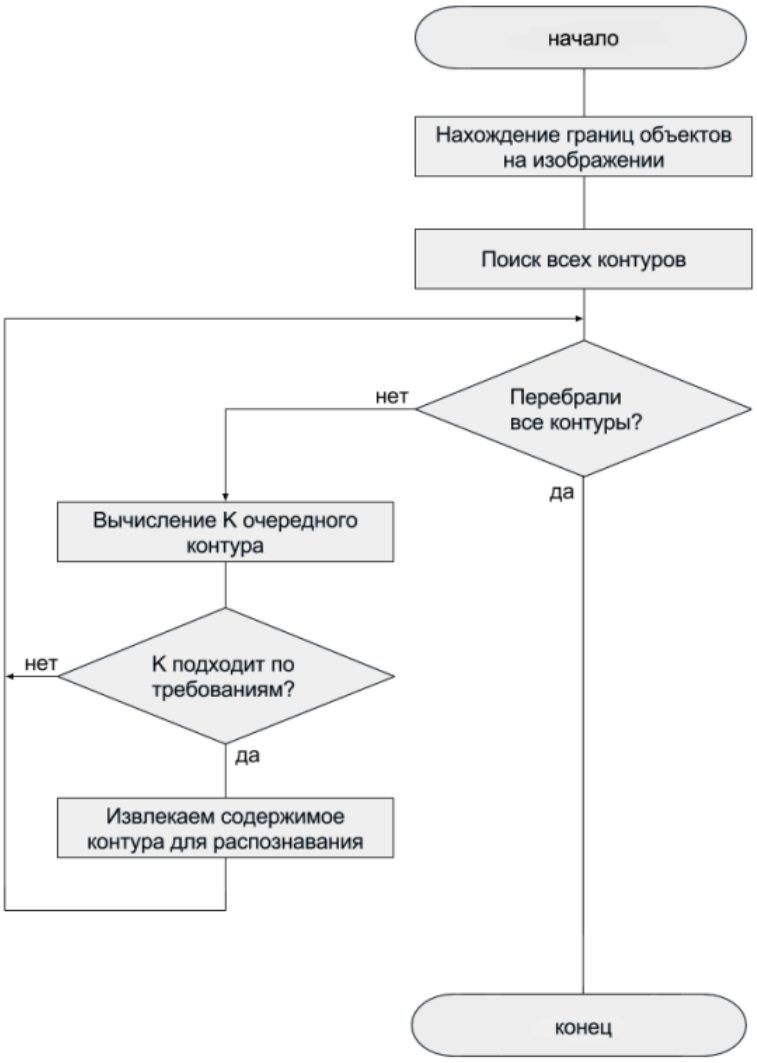

Рис.11-Алгоритм детектирования одного кадра

1. Предобработка изображения

В качестве предобработки, изображение было переведено в градации серого и была повышена контрастность изображения. Повышение контрастности изображения позволяет более ярко выделить границы объектов и различные детали на изображении. В свою очередь более четкие и яркие границы объектов позволяют более точно вычислять геометрические характеристики и форму объектов, что улучшает качество детектирования.

Для повышения контрастности было произведено выравнивание гистограмм яркости изображения.

Гистограмма изображения — это графическое представление распределения интенсивности изображения. Она количественно определяет количество пикселей для каждого рассматриваемого значения интенсивности.

Выравнивание гистограммы — это метод, который улучшает контрастность изображения, чтобы расширить диапазон интенсивности. Этот метод реализован в библиотеке OpenCV [11].

Посмотрим на выравнивание гистограмм на примере.

Исходное изображение и его гистограмма представлены на рисунке 24, 25.

Рис.12- исходный кадр дорожной сцены

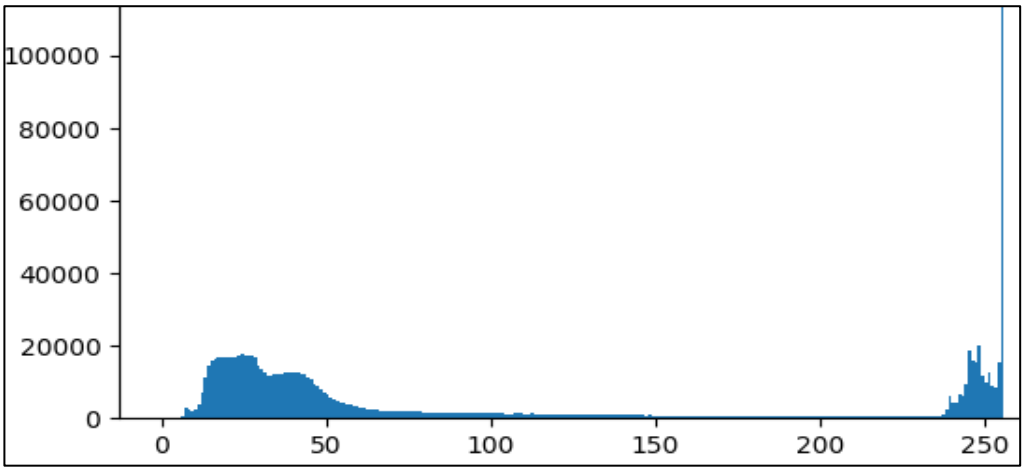

Рис.13- Гистограмма исходного изображения

После применения функции выравнивания гистограммы, реализованной в библиотеке OpenCV получены следующие результаты (рисунок 14, 15):

Рис.14- кадр дорожной сцены с повышенным контрастом

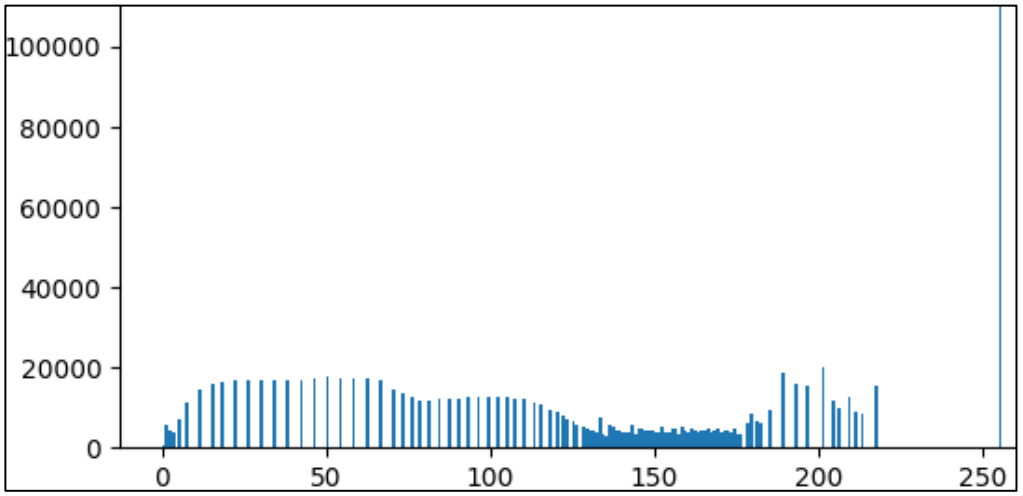

Рис.15- Гистограмма обработанного изображения

Также упрощения вычислительных процессов алгоритма и частичного удаления шумов (деревья, дома), применена обработка изображения в пространстве HSV.

С помощью данной обработки на изображении частично устраняется задний фон знаков, небо и др. объекты, загружающие алгоритм локализации (даже изображение внутри знака может быть удалено, важно сохранить только внешний контур). Для этого были подобраны пороговые значения составляющих пространства HSV. Результат представлен на рисунке 16.

- Hue-3
- Saturation-20
- Value-20

Рис.16- Удаление шумов и неинформативных объектов

2. Поиск контуров

Поиск контуров производится при помощи метода границ Кэнни. Результат обработки подготовленного изображения после применения алгоритма Кэнни для поиска границ можно наблюдать на рисунке 17. Полученное изображение является

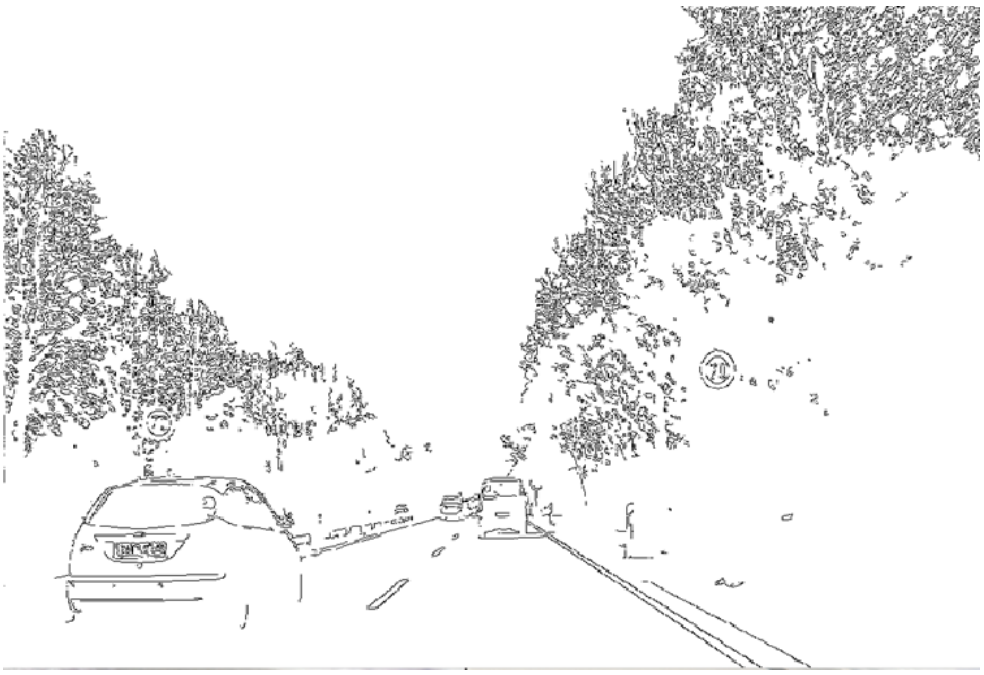

Рис.17- Детектор границ Кэнни

После применения метода границ Кэнни необходимо получить набор замкнутых контуров. Это делается с помощью функции «findContours» библиотеки OpenCV. Эта функция возвращает целое множество контуров, и каждому контору сопоставлен массив numpy с координатами точек этого контура. Таким образом, после работы функции имеется контейнер, содержащий наборы точек, которыми можно приблизить замкнутые кривые.

3. Проверка контуров на принадлежность к дорожным знакам

Что бы определить, является ли найденный контур дорожным знаком, надо обратить внимание на геометрические характеристики знаков. В большинстве своем знаки имеют три вида форм: круглые, треугольные и квадратные.

Соответственно, нужно найти контуры, форма которых подходит под характеристику знаков, но не зависит от масштаба, т.к. знак в кадре может находиться на разном расстоянии и под разным углом осмотра. Такой характеристикой обладает коэффициент K, который представляет собой отношение площади контура к квадрату его периметра.

$$K = \frac{S}{P^2} \qquad (5.1)$$

Где S-площадь замкнутого контура, P-периметр контура.

Например, коэффициент К для квадратной формы равен $K = \frac{a^2}{(4a)^2} = 0.0625$. Как видно из примера, К не зависит от сторон фигур (максимум от их соотношений). В реальных изображениях К варьируется в некотором диапазоне из-за изменения плоскости дорожного знака относительно камеры.

Экспериментальным путем были вычислены диапазоны К для фигур (таблица 3):

Таблица 3 - диапазоны К для разных геометрических фигур

Вид формы знака	K_{min}	K_{max}
Эллипс (круг)	0.06	0.1
Прямоугольник (квадрат)	0.05	0.065
треугольник	0.020	0.055

Таким образом, для каждого контура будет рассчитан коэффициент К и сопоставлен сданными из таблицы 5.2. Если коэффициент К подходит под значения, то можно считать найденный контур окантовкой дорожного знака.

Пример поиска контуров приведены на рисунке 18.

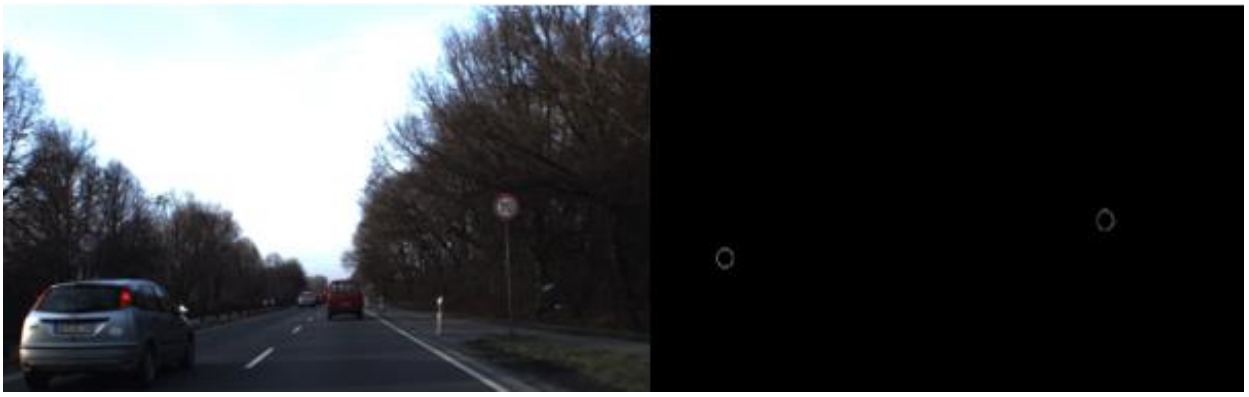

Рис.18- Найденные контуры

4. Отрисовка ограничивающей области (bounding box)

Когда найденные контуры прошли проверку и выявлены контуры, содержащие знаки, необходимо отобразить ограничивающий прямоугольник на входном изображении вокруг найденного контура (рисунок 19).

Рис.19- Результат работы детектора

Результаты и оценка качества

Результаты работы алгоритма приведены в таблице 4

Таблица 4- Показатели метрик качества алгоритма

	Accuracy	Precision	Recall	F1-score	FPS
Изображения (300 шт.)	0.18	0.11	0.24	0.15	---
Видеопоток (Длина-15с)	0.08	0.09	0.88	0.167	17

По полученным данным можно сделать вывод о весьма малой точности алгоритма. Несмотря на стабильность работы, частота нахождения знака оставляет желать лучшего. Алгоритм начинает испытывать проблемы при насыщенном контурами заднем фоне, при поиске треугольных знаков, а также большой проблемой является отсечение лишних областей.

Пример некачественной работы алгоритма детектирования приведен на рисунке 20

Рис.20- Пример некачественной работы детектора

3.3 Обучение каскада классификаторов

В качестве модели детектора был взят и обучен предобученный каскад Хаара, который обучался на примерах дорожных знаках.

Для дообучения каскада Хаара в OpenCV были выполнены следующие шаги:

1. Создание позитивных и негативных изображений:

Позитивные (рис.21а) и негативные (рис.21б) изображения. Позитивные изображения содержат объекты, которые вы хотите распознавать, а негативные - изображения без объектов. Размер положительной выборки составил 300 изображений, а отрицательной- 500 изображений

Рис.21- Примеры из обучающей выборки

а) положительные б) отрицательные

2. Создание файла описания позитивных изображений.

В этом файле указывается количество позитивных изображений, их размеры и путь к каждому изображению.

3. Создание файла описания негативных изображений.

В этом файле указывается количество негативных изображений и путь к каждому изображению.

4. Обучение каскада Хаара.

После подготовки данных можно приступить к обучению каскада Хаара с помощью функции cv2.CascadeClassifier.train(). В процессе обучения будет создан XML-файл, который содержит информацию о каскаде Хаара.

- Количество уровней каскада для обучения- 18.
- Время обучения – 27ч.

Результаты и оценка качества

Результаты работы алгоритма на изображениях приведены в таблице 5

Таблица 5- Показатели метрик качества алгоритма на изображениях

	Accuracy	Precision	Recall	F1-score
Изображения (300 шт.) (предобученный каскад)	0.67	0.72	0.58	0.640
Изображения (300 шт.) (дообученный каскад)	0.72	0.76	0.62	0.675

Время, необходимое для поиска дорожных знаков одним каскадом с использованием на изображении размера 1280 × 720 при параметрах scaleFactor = 1.1; minSize = 30 × 30 ; maxSize = 70 × 70 составляет 540-580 мс;

Результаты работы алгоритма на видеопотоке приведены в таблице 6

Таблица 6 - Показатели метрик качества алгоритма на видеопотоке

Величина сканирующего окна	Accuracy	Precision	Recall	F1-score	FPS
1.1	0.88	0.93	0.86	0.89	6
1.2	0.9	0.9	0.9	0.9	9

1.3	0.91	0.9	0.94	0.91	14

По полученным данным можно сделать вывод о хорошей точности алгоритма. На изображении алгоритм допускает больше ошибок, однако при работе с видеопотоком алгоритм показывает большую точность, т.к. если он не нашел знак на одном кадре, то он может обнаружить его на следующем. По сравнению с детектором Кэнни у этого алгоритма качество детектирования много выше.

Что касается частоты кадров (FPS) при работе алгоритма с видеопотоком, то такая частота вряд ли применима к автомобильным транспортным средствам. На скорости ≥ 30км/ч алгоритм не успеет обработать кадр вовремя. Для применения системы в автомобилях детектор должен выдавать минимум 30 FPS.

Однако, такая система может найти применение, например, в автономных устройствах доставки от «Яндекса» или «Почты России», которые передвигаются по городу с небольшой скоростью и алгоритм без затруднения будет успевать обрабатывать поток видео.

3.4 Выводы по главе

В первом разделе главы были представлены данные для реализации детекторов.

Во втором разделе главы реализован детектор, основанный на методе границ Кэнни. Проведена оценка качества системы.

В третьем разделе реализован детектор, основанный на методе Виолы-Джонса. Проведена оценка качества системы.

Глава 4. ОБУЧЕНИЕ И ТЕСТИРОВАНИЕ НЕЙРОННЫХ СЕТЕЙ

В данной главе построена и обучена на заданном наборе данных модель СНС. Вычислены точности классификации с применением аугментации и без.

Реализация обучения произведена на языке Python в интерактивной среде Google Colab, в которой есть возможность подключить графический процессор (GPU) для получения дополнительных мощностей и, соответсвенно, более быстрого обучения сети.

Для построения архитектуры сети используется библиотека Keras [12]. Эта библиотека позволяет создавать и обучать нейронные сети с помощью простого и интуитивно понятного интерфейса на языке Python.

4.1 Данные для обучения и тестирования

(German Traffic Sign Recognition Benchmark) [13]

Данный набор содержит 43 класса и разбит на две части: тренировочную и тестовую выборки состоящих из 39209 и 12631 изображений соответственно. Набор данных для тестирования содержит 12631 изображений, которые не принимают участия в обучении СНС.

Все виды классов приведены на рисунке 22.

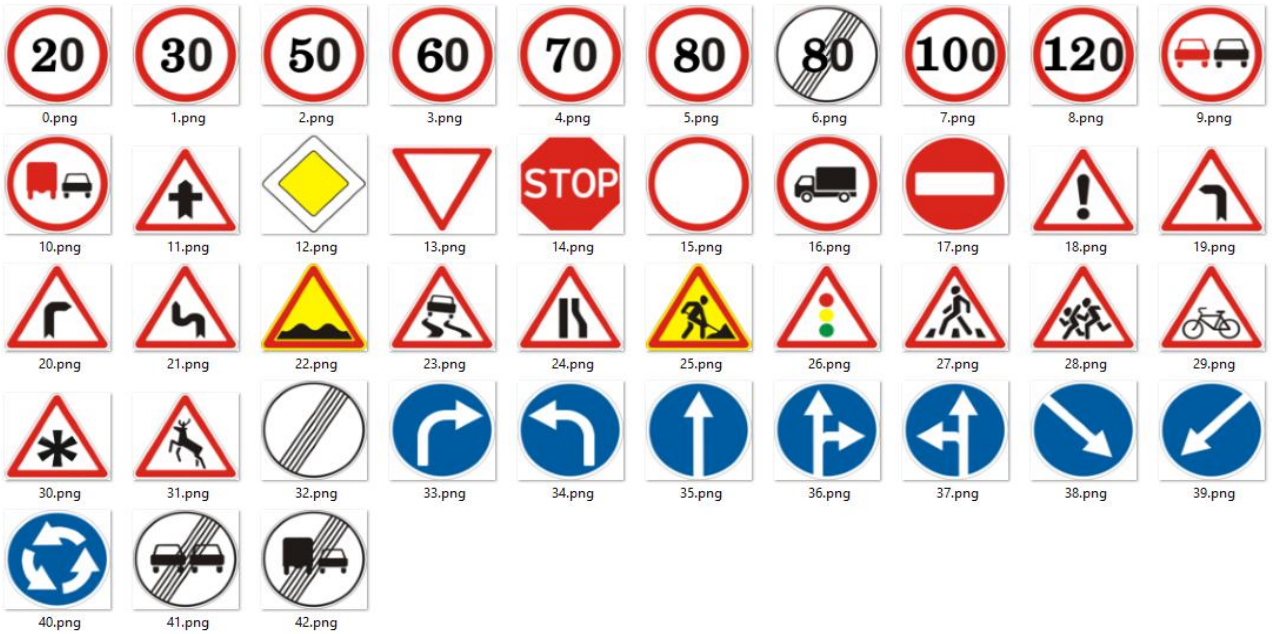

Рис. 22- Примеры классов дорожных знаков

4.2 Предобработка данных

Данные распределяются на тренировочную и валидационную выборки по 80% и 20%, соответственно. Тестовая выборка для проверки обученной модели состоит из 12605 изображений и отдельно хранится в датасете. На тренировочной выборке модель обучается. На валидационной наблюдается текущая ошибка и точность классификации. По этим критериям принимается решение об улучшении, модификации модели.

Каждое изображение приводится к размеру 30 x 30. Изображения нормируются по каждому каналу.

В качестве разведочного анализа выведем гистограмму распределения количества изображений в обучающем наборе для каждого класса дорожных знаков (рисунок 23).

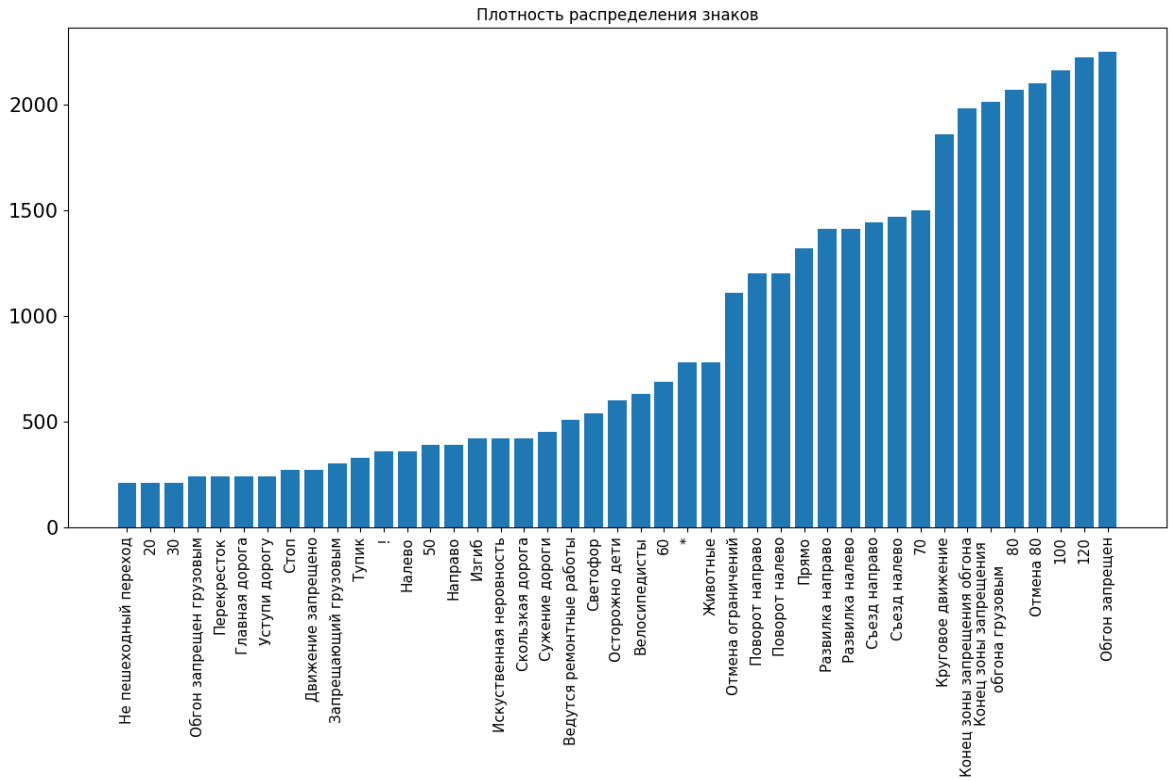

Рис. 23– Гистограмма распределения дорожных знаков по классам

Из гистограммы видно, что данные сильно не сбалансированы, что может негативно сказаться на качестве обучения, поэтому для тренировочной выборки будут применены операции аугментации для классов с малым количеством обучающих примеров (<700).

Аугментация представляет собой процесс увеличения или улучшения данных путем добавления новых элементов или свойств. В машинном обучении аугментация используется для улучшения качества данных путем создания измененных копий существующих данных, которые могут быть использованы для обучения модели. Это может включать различные аффинные преобразования (изменение размера, поворот, смещение), изменение цвета и другие преобразования изображений. В результате аугментации можно получить более разнообразный набор данных, что может привести к улучшению точности модели и ее способности к обобщению.

После применения аугментации распределение выглядит следующим образом (рисунок 24):

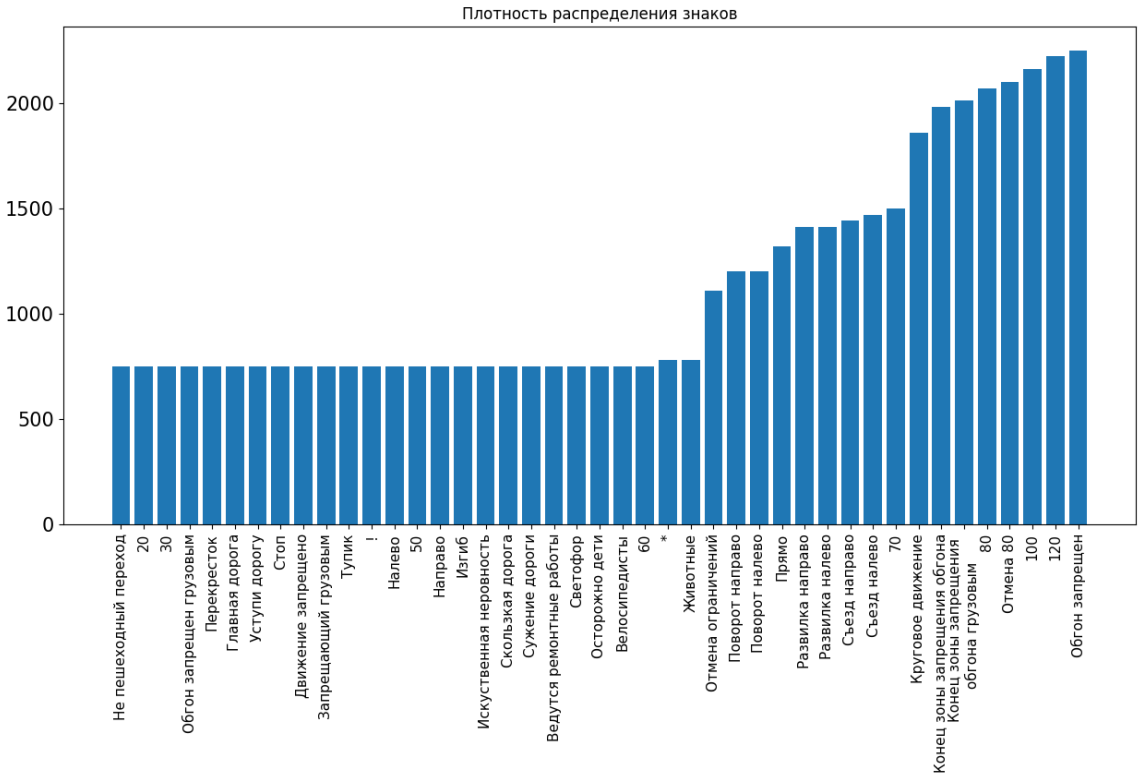

Рис. 24– Гистограмма распределения дорожных знаков по классам После аугментации

4.3 Описание архитектуры сверточной нейронной сети

В соответствии со статьей [14] была выбрана следующая архитектура СНС, которая показала высокую точность распознавания (≈ 95% – 96%) .

Архитектура модели (рисунок 25):

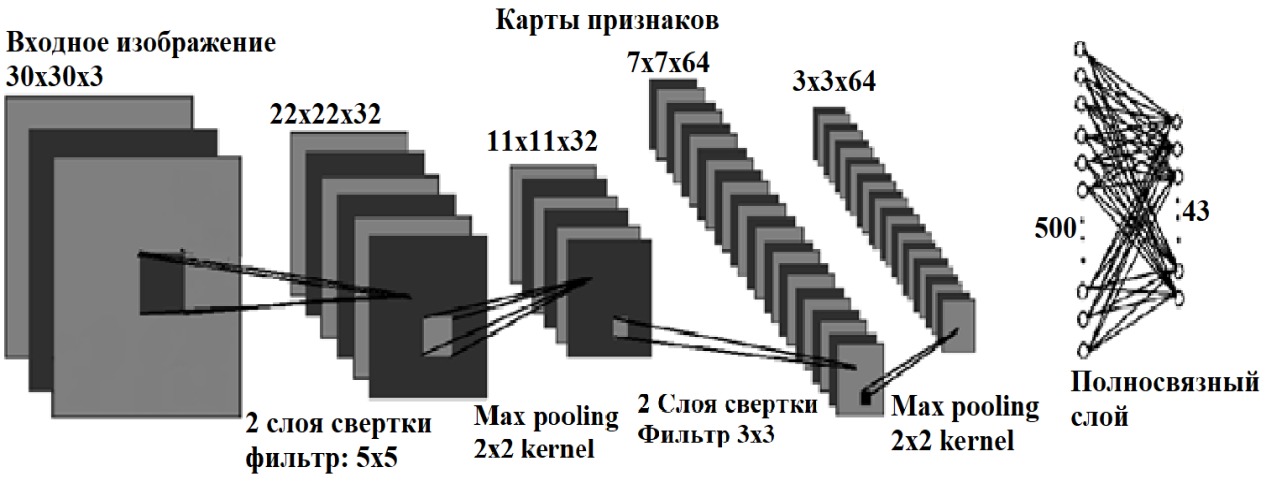

Рис 25 – Архитектура построенной сверточной нейронной сети

- 2 слоя свертки по 32 фильтра в каждом. Размером фильтра 5x5, функция активации – «relu»
- Слой объединения (pooling) с размером окна 2x2
- Слой Dropout (исключение 25% нейронов)
- 2 слоя свертки по 64 фильтра в каждом. Размером фильтра 3x3, функция активации – «relu»
- Слой объединения (pooling) с размером окна 2x2
- Слой Dropout (исключение 25% нейронов)
- Слой Flatten, чтобы сжать слои в 1 измерение
- Полносвязный слой (Dense), состоящий из 500 нейронов. Функция активации – «relu»
- Слой Dropout (исключение 50% нейронов)
- Полносвязный слой (Dense), состоящий из 43 нейронов. Функция активации – «softmax»

4.4 Обучение сети без аугментаций

Параметры обучения:

- Функция потерь – кросс-энтропия

- Оптимизатор – Adam
- Размер батчей – 64
- Количество эпох – 75

График обучения представлен на рисунке 26.

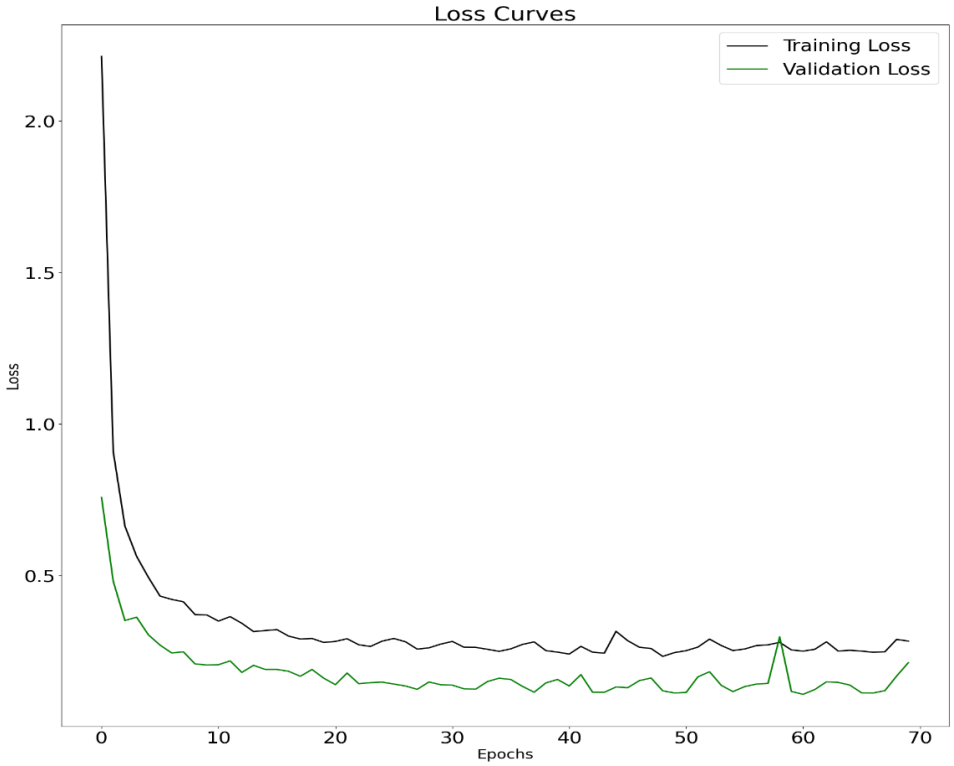

Рис.26– График обучения сети

Как видно из графика, к 50 эпохе модель перестала значительно обучаться, а валидационная ошибка после 60 эпохи даже начала показывать тенденцию к росту.

Ошибки в конце обучения:

- Общая ошибка (loss) = 28.1%
- Точность (accuracy) = 92.3%
- Ошибка на валидации(val_loss) = 21.25%
- Точность валидаци (val_accuracy) = 94%.
- Время обучения сети 5.5 мин.

4.5 Результаты

Для оценки обучения нейронной сети были импортированы изображения из тестирующего набора данных. Для тестовых данных были рассчитаны отчет по

классификации (для каждого класса рассчитаны метрики качества классификации (precision, recall, f1-score)), а также матрица ошибок, которая была представлена в виде тепловой карты.

Так же несколько случайных примеров тестовой выборки были прогнаны через нейросеть для визуальной демонстрации распознавания (рисунок 27)

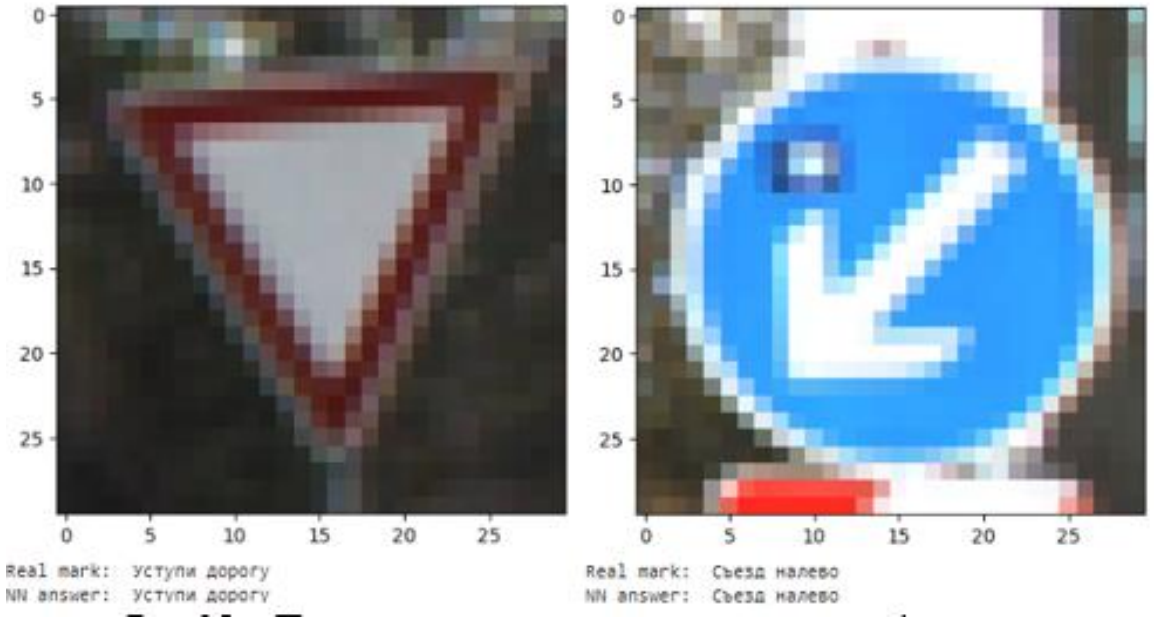

Рис.27 – Пример распознавания на тестовых данных

1. Отчет по классификации (рисунок 28).

	precision	recall	f1-score	support
20	0.60	1.00	0.75	60
30	0.96	0.97	0.96	720
50	0.98	0.98	0.98	750
60	0.91	0.96	0.93	450
70	0.99	0.97	0.98	660
80	0.96	0.91	0.94	630
Отмена 80	0.93	0.88	0.90	150
100	0.98	0.96	0.97	450
120	0.97	0.95	0.96	450
Обгон запрещен	0.94	1.00	0.97	480
Обгон запрещен грузовым	1.00	0.98	0.99	660
Перекресток	0.99	0.90	0.95	420
Главная дорога	0.97	0.90	0.93	690
Уступи дорогу	0.98	0.98	0.98	720
Стоп	0.95	0.99	0.97	270
Движение запрещено	0.94	0.99	0.96	210
Запрещающий грузовым	0.90	1.00	0.95	150
Тупик	1.00	0.81	0.90	360
!	0.99	0.81	0.89	390
Налево	0.90	1.00	0.94	60
Направо	0.70	1.00	0.82	90

Изгиб	0.76	0.71	0.74	90
Искуственная неровность	0.96	0.93	0.94	120
Скользкая дорога	0.67	0.97	0.79	150
Сужение дороги	0.85	0.93	0.89	90
Ведутся ремонтные работы	0.99	0.94	0.96	480
Светофор	0.91	0.88	0.90	180
Не пешеходный переход	0.66	0.87	0.75	60
Осторожно дети	0.97	0.99	0.98	150
Велосипедисты	0.78	0.97	0.86	90
*	0.80	0.68	0.74	150
Животные	0.91	0.99	0.95	270
Отмена ограничений	0.55	1.00	0.71	60
Поворот направо	0.99	0.97	0.98	210
Поворот налево	0.94	1.00	0.97	120
Прямо	0.99	0.97	0.98	390
Развилка направо	0.98	0.92	0.95	120
Развилка налево	0.98	0.98	0.98	60
Съезд направо	1.00	0.93	0.96	690
Съезд налево	0.82	0.73	0.78	90
Круговое движение	0.82	0.92	0.87	90
Конец зоны запрещения обгона	0.75	0.92	0.83	60
Конец зоны запрещения обгона грузовым	0.64	0.94	0.76	90
accuracy			0.94	12630
macro avg	0.89	0.93	0.90	12630
weighted avg	0.95	0.94	0.94	12630

Рис.28 – Отчет по классификации

Время классификации одного предобработанного изображения: 0.85 – 1 мс

По данному отчету можно сказать о неплохом качестве обучения, показатели метрик приемлемы для большинства классов, а показатели Accuracy находятся в районе ≥ 90%.

2. Матрица ошибок (рисунок 29)

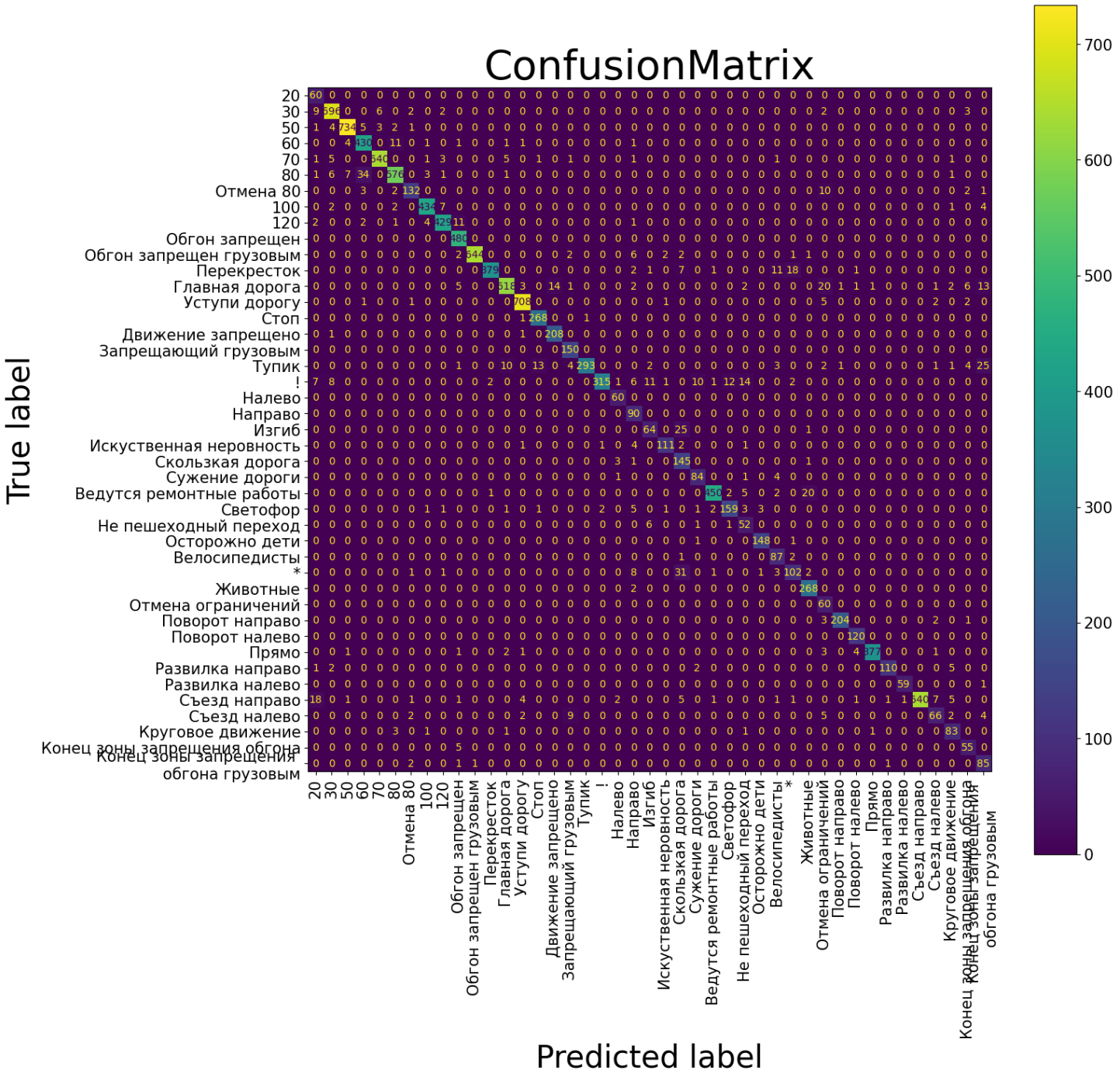

Рис. 29 – Матрица ошибок

По матрице ошибок так же можно сказать о неплохих показателях распознавания.

4.6 Обучение нейронных сетей с аугментациями

Параметры обучения:

- Функция потерь – кросс-энтропия
- Оптимизатор – Adam

- Размер батчей – 64
- Количество эпох – 72

График обучения представлен на рисунке 30.

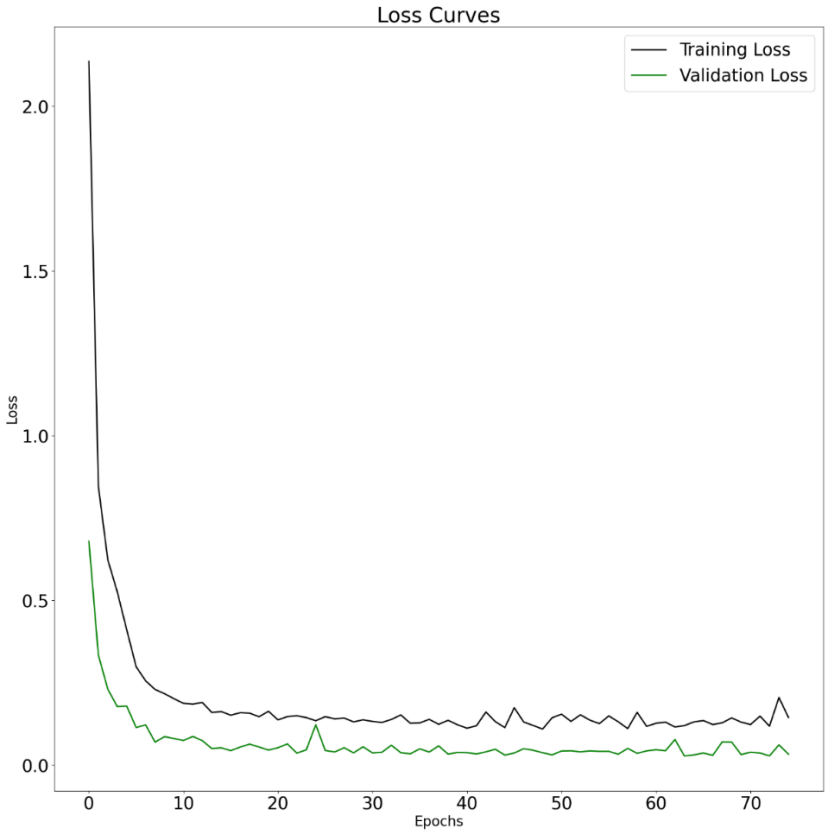

Рис.30- График обучения сети

Как видно из графика, примерно к 40 эпохе модель перестала значительно обучаться и вышла на плато, а валидационная ошибка после 65 эпохи даже начала показывать тенденцию к росту.

Ошибки в конце обучения:

- Общая ошибка (loss) = 14.4%
- Точность (accuracy) = 94.3%
- Ошибка на валидации(val_loss) = 3.25%
- Точность валидаци (val_accuracy) = 97%.
- Время обучения сети 6.5 мин

4.7 Результаты

Для оценки обучения нейронной сети были импортированы изображения из тестирующего набора данных. Для тестовых данных были рассчитаны отчет по классификации (для каждого класса рассчитаны метрики качества классификации (precision, recall, f1-score)), а также матрица ошибок, которая была представлена в виде тепловой карты.

Так же несколько случайных примеров тестовой выборки были прогнаны через нейросеть для визуальной демонстрации распознавания (рисунок 31).

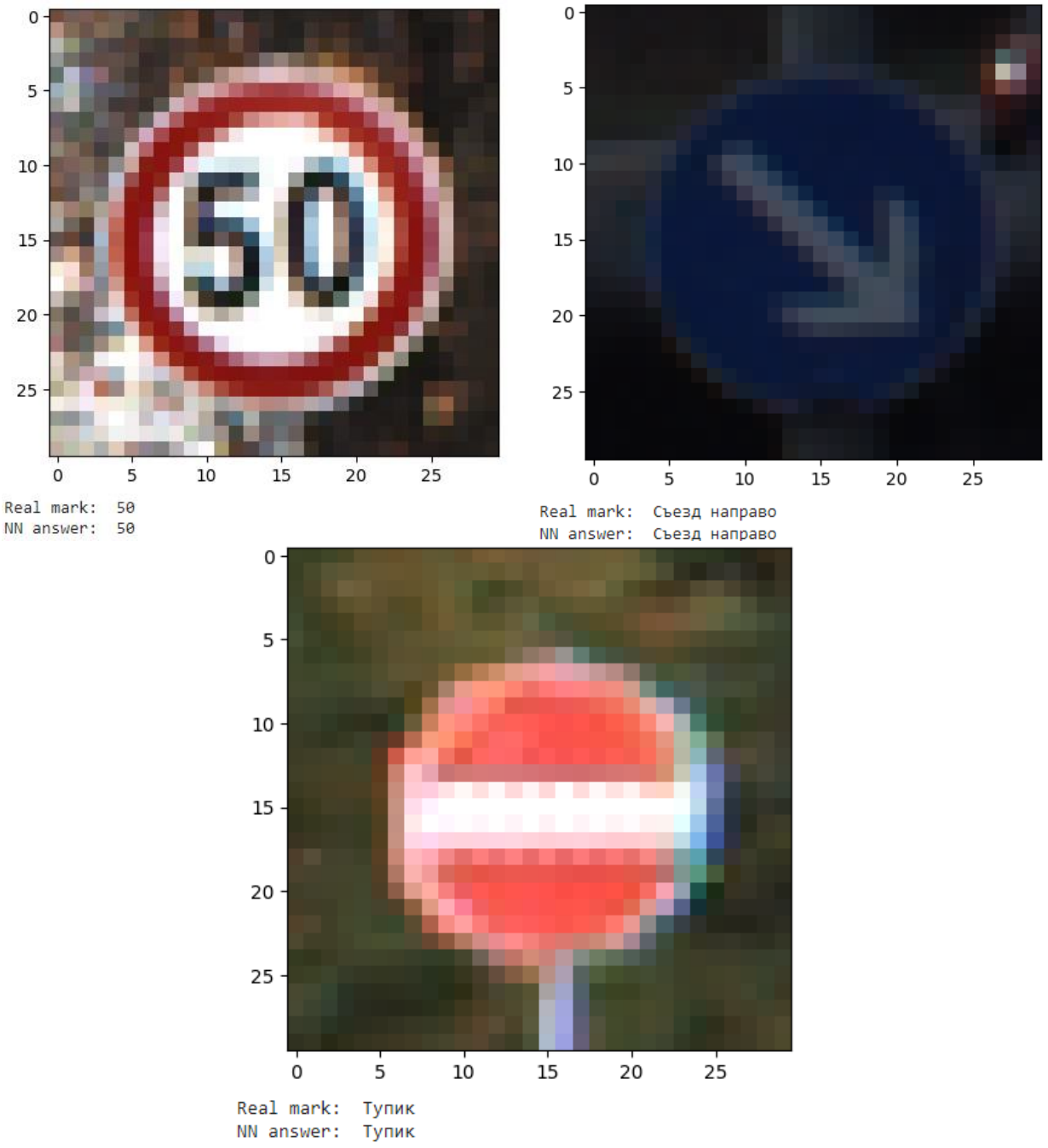

Рис.31 – Пример распознавания на тестовых данных

1. Отчет по классификации (рисунок 32).

	precision	recall	f1-score	support
20	0.98	0.98	0.98	60
30	0.95	0.98	0.96	720
50	0.97	0.98	0.98	750
60	0.93	0.96	0.95	450
70	0.98	0.97	0.97	660
80	0.91	0.97	0.94	630
Отмена 80	0.95	0.85	0.90	150
100	0.97	0.98	0.97	450
120	0.96	0.93	0.94	450
Обгон запрещен	0.96	1.00	0.98	480
Обгон запрещен грузовым	0.98	0.99	0.98	660
Перекресток	0.95	0.95	0.95	420
Главная дорога	0.98	0.93	0.96	690
Уступи дорогу	0.98	0.99	0.98	720
Стоп	0.99	0.99	0.99	270
Движение запрещено	0.98	0.97	0.97	210
Запрещающий грузовым	0.99	0.95	0.97	150
Тупик	1.00	0.99	1.00	360
!	0.96	0.80	0.87	390
Налево	1.00	0.72	0.83	60
Направо	0.86	0.97	0.91	90
Изгиб	0.89	0.81	0.85	90
Искуственная неровность	0.97	0.95	0.96	120
Скользкая дорога	0.77	0.97	0.86	150
Сужение дороги	0.70	0.90	0.79	90
Ведутся ремонтные работы	0.91	0.98	0.95	480
Светофор	0.81	0.82	0.82	180
Не пешеходный переход	0.89	0.55	0.68	60
Осторожно дети	0.87	0.99	0.92	150
Велосипедисты	0.93	0.92	0.93	90
*	0.92	0.73	0.81	150
Животные	0.93	0.99	0.96	270
Отмена ограничений	0.94	1.00	0.97	60
Поворот направо	0.94	0.98	0.96	210
Поворот налево	0.98	0.99	0.98	120
Прямо	0.99	0.96	0.97	390
Развилка направо	0.98	0.99	0.98	120
Развилка налево	0.89	0.98	0.94	60
Съезд направо	0.99	0.94	0.97	690
Съезд налево	1.00	0.84	0.92	90
Круговое движение	0.80	0.80	0.80	90
Конец зоны запрещения обгона	0.98	0.87	0.92	60
Конец зоны запрещения обгона грузовым	0.96	1.00	0.98	90
accuracy			0.95	12630
macro avg	0.94	0.93	0.93	12630
weighted avg	0.95	0.95	0.95	12630

Рис.32 – Отчет по классификации

Время классификации одного предобработанного изображения: 0.9 – 1мс

По данному отчету можно сказать о значительном улучшении модели после аугментации обучающей выборки, показатели метрик высоки для большинства классов, а показатели Accuracy находятся в районе 95%

2. Матрица ошибок (рисунок 33).

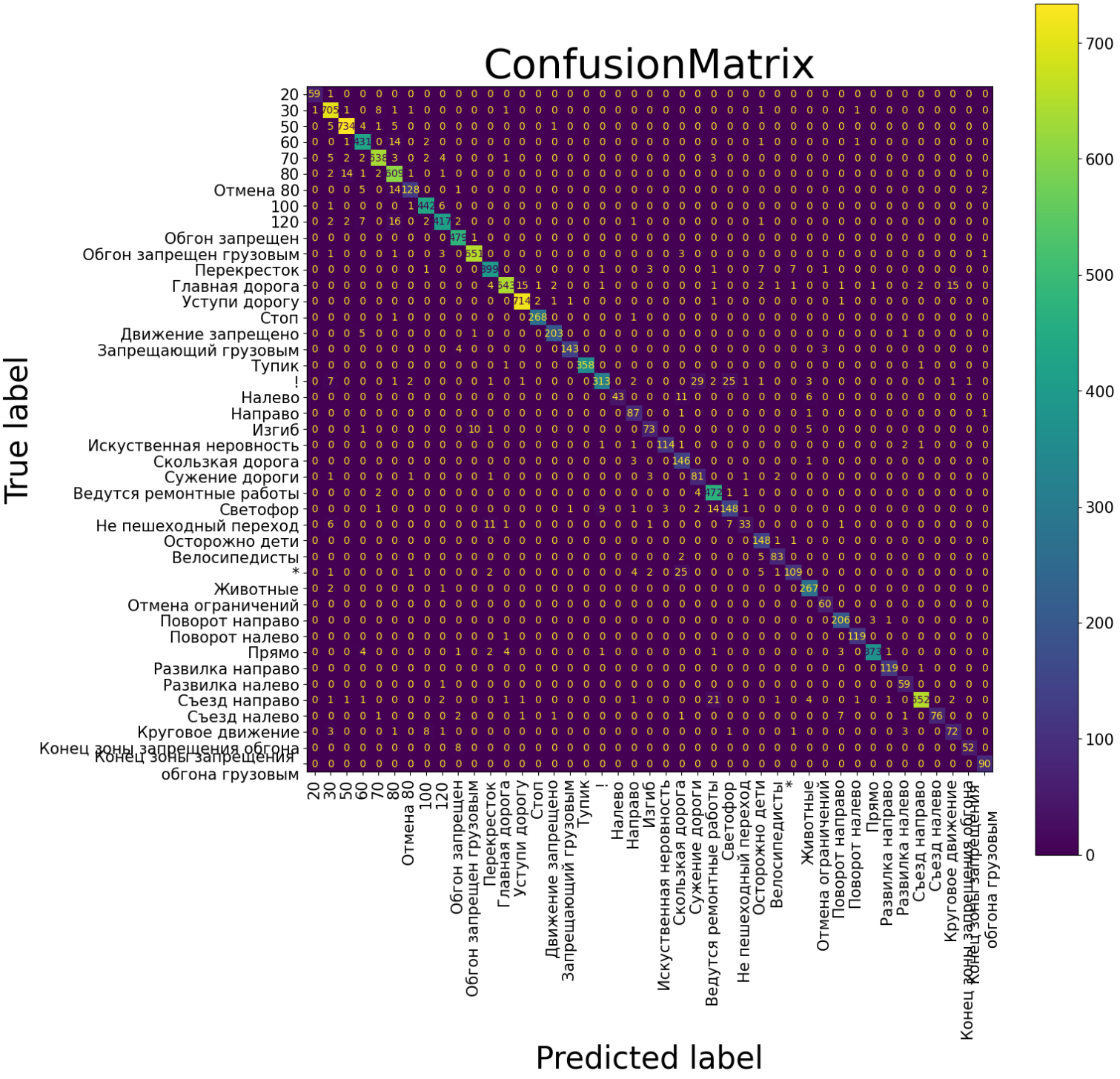

Рис. 33 – Матрица ошибок

По матрице ошибок так же можно сказать о высоких показателях распознавания.

4.8 Метод борьбы с неинформативными областями, полученными на этапе локализации.

Методы детектирования, как и любые другие методы не идеальны и могут допускать ошибки. Для потенциального повышения качества детектирования можно использовать дополнительные этапы проверки для уточнения, находится ли нужный объект внутри выделенной области (Bounding box). На этом этапе нет необходимости классифицировать объект внутри окна, а лишь сделать вывод, есть ли в нем объект распознавания или нет. Если вывод положителен, то изображение подается дальше на распознавание, в ином случае выделенная область не обрабатывается. И одним из способов такой проверки является использование сверточного автоэнкодера.

4.8.1 Сверточные автоэнкодеры

Сверточный автоэнкодер [3] — это нейронная сеть, которая используется для сжатия до базовых паттернов и восстановления в исходное состояние изображений. Он состоит из двух частей: энкодера и декодера. Энкодер преобразует входное изображение в скрытое представление, а декодер восстанавливает изображение из этого представления. Такая архитектура связана с тем, что в процессе обучения сверточный автоэнкодер учится выделять наиболее важные признаки изображения и сохранять их в скрытом представлении (рисунок 34).

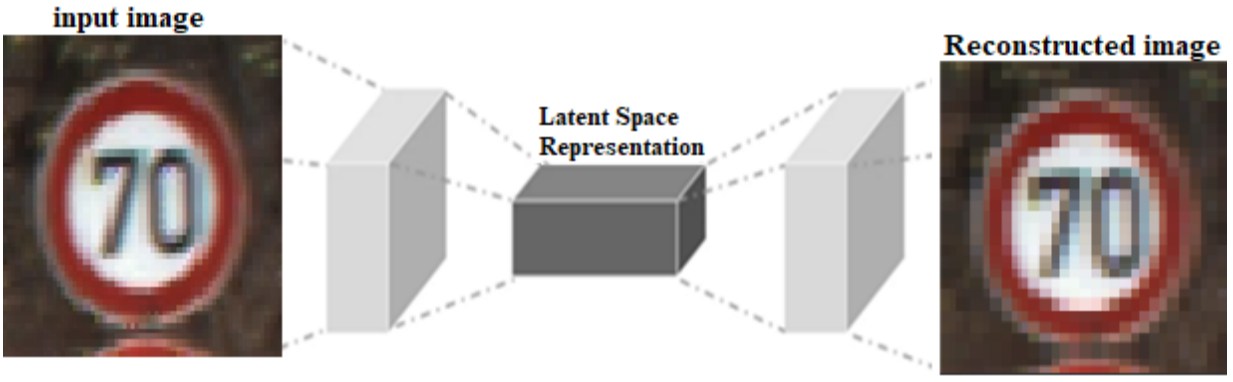

Рис.34 – Сверточный автоэнкодер

Для проверки локализованной области на наличие объекта при помощи сверточного автокодировщика необходимо выполнить следующие шаги:

1. Обучение сверточного автоэнкодера на наборе данных из объектов, которые представляют собой положительные объекты. Для этого выбираются изображения объектов и обучается сверточный автоэнкодер на их основе.

   В данной работе набор данных должен состоять из множества знаков, которые планируется распознавать. Для этого можно использовать тот же набор данных, что и для обучения сверточной нейронной сети. Автоэнкодеру не нужны метки классов, только сами изображения.
2. Извлечение признаков из изображений, на которых обучался автоэнкодер. Для этого, после того как сработал энкодер и перед обратным декодированием, нужно взять слой из модели, который отвечает за кодирование изображения и сохранить его. Таким образом, сохраняться все кодированные представления изображения. [15]
3. Обученный энкодер сохраняется отдельно. Декодер был нужен только для обучения сети.
4. Проверка детектирования объектов. Для этого область изображения с потенциально найденным объектом прогоняется через обученный энкодер. Получаемое скрытое представление сравнивается с кодированными представлениями изображений из обучающей выборки. Сравнение производится при помощи математических метрик, таких как Евклидово или косинусное расстояние. Если вычисленное расстояние ниже заданного порога, то в закодированных представлениях из п.2. найдено такое изображение, которое похоже на изображение, поданное на вход энкодера, и детектирование можно считать успешным.

Таким образом, использование сверточного автоэнкодера для проверки детектирования может повысить качество работы системы и уменьшить количество ошибок.

4.8.2 Обучение сверточного автоэнкодера

Для обучения автокодировщика были использованы те же инструменты и выборки, что и для обучения сверточной нейронной сети.

Архитектура сверточного автокодировщика (рисунок 35):

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 224, 224, 32)      896
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 112, 112, 32)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 112, 112, 64)      18496
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 56, 56, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 56, 56, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 28, 28, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 28, 28, 256)       295168
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 14, 14, 256)       0
_________________________________________________________________
flatten (Flatten)            (None, 50176)             0
_________________________________________________________________
dense (Dense)                (None, 32)                1605664
=================================================================
Total params: 1,994,080
Trainable params: 1,994,080
Non-trainable params: 0
_________________________________________________________________
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_1 (Dense)              (None, 50176)             1655808
_________________________________________________________________
reshape (Reshape)            (None, 14, 14, 256)       0
_________________________________________________________________
conv2d_transpose (Conv2DTran (None, 28, 28, 128)       295040
_________________________________________________________________
conv2d_transpose_1 (Conv2DTr (None, 56, 56, 64)        73792
_________________________________________________________________
conv2d_transpose_2 (Conv2DTr (None, 112, 112, 32)      18464
_________________________________________________________________
conv2d_transpose_3 (Conv2DTr (None, 224, 224, 3)       867
=================================================================
```

Рис.35 -Архитектура сверточного автокодировщика

Параметры обучения:

- Функция потерь – среднеквадратичная ошибка
- Оптимизатор – «adamax»

- Количество эпох – 40
- Время обучения – 148 мин.

В качестве метрики расстояния было выбрано косинусное расстояние.

Для наглядной демонстрации работы сети были написаны функция для расчета расстояния и отображения трех ближайших изображений.

Пример работы (рисунок 36):

Цвета на изображениях отображены не совсем корректно из-за представления изображений не в формате RGB, а в формате BGR (синий цвет на самом деле красный). Но это никак не влияет на качество выполнения программы.

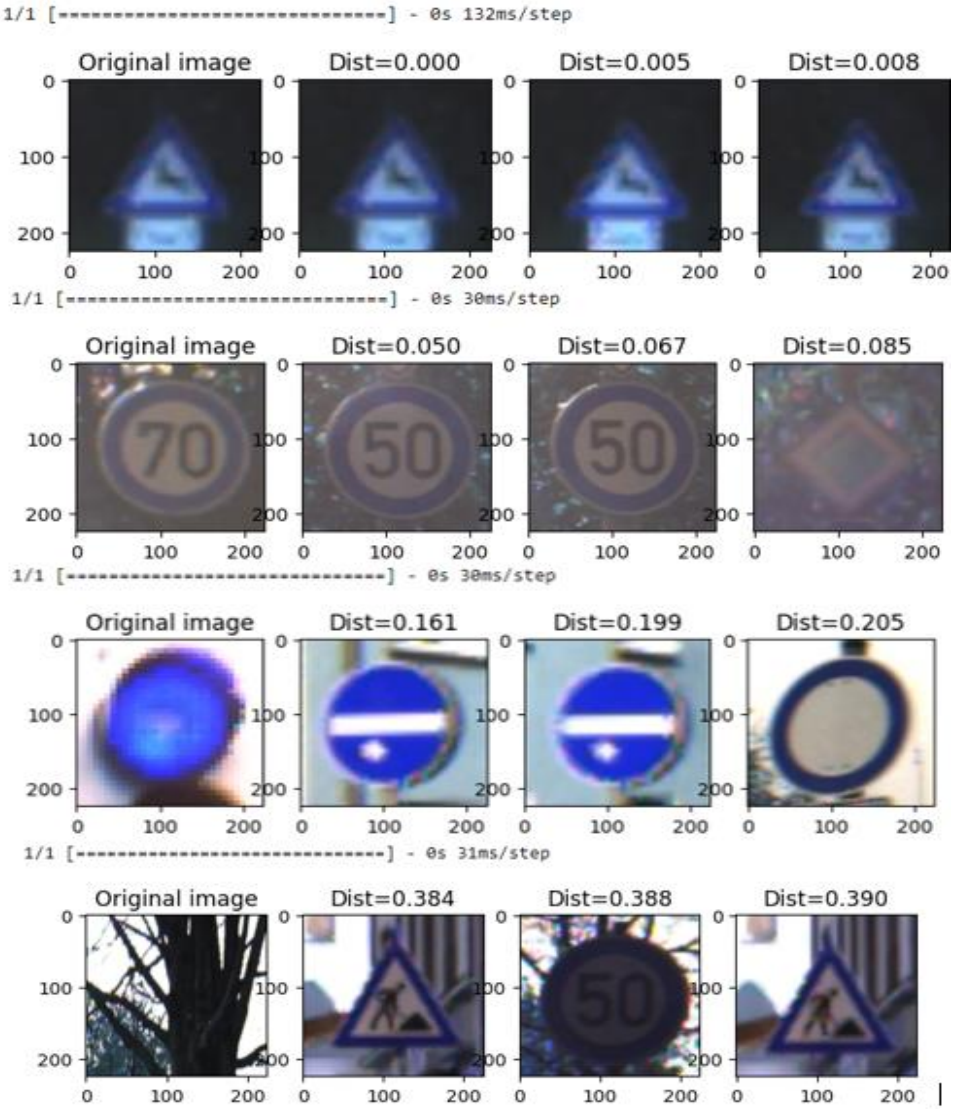

Рис.36- Отображение похожих изображений

Как видно из примеров работы программы, изображения, на которых содержатся знак действительно находят похожее изображение с малым

показателем расстояния, даже если их классы не совпадают и имеются незначительные отличия в виде знака. В то время как объекты, не содержащие знак, сильно отличаются от ближайших похожих изображений.

4.9 Выводы по главе

В первом разделе главы были представлены данные для обучения и тестирования.

Во втором разделе главы представлена предварительная обработка данных для обучения.

В третьем разделе главы построена архитектуры модели CHC.

В четвертом и пятом разделах главы проведено обучение архитектуры без аугментации и рассчитаны показатели качества.

В шестом и седьмом разделах главы проведено обучение архитектур с аугментацией, сравнение результатов и точности на тестовой выборке.

В восьмом и девятом разделах главы был построен и обучен сверточный автоэнкодер для промежуточной проверки детектированных знаков.

Глава 5. АНАЛИЗ И СРАВНЕНИЕ КОНЕЧНЫХ СИСТЕМ РАСПОЗНАВАНИЯ

В этой главе будут рассмотрены системы автоматического распознавания дорожных знаков, произведена оценка их качества и сравнение друг с другом.

5.1 Пример работы разработанных систем распознавания.

Для реализации конечной системы были сохранены обученные модели нейросетей и импортированы в программу детектирования. Для этого были разработаны 2 модуля, где импортируются обученные нейронные сети, и происходит предобработка изображения для подачи на вход сети. Результат выхода классификатора с помощью функции cv2:putText отображался на выходном изображении/кадре.

Пример работы системы, построенной на каскаде классификаторов, можно наблюдать на рисунке 37,38.

Рис.37- Работа системы, построенной на каскаде классификаторов

Рис.38- Работа системы, построенной на каскаде классификаторов

Пример работы системы, построенной на афинных признаках знаков, можно наблюдать на рисунке 39,40.

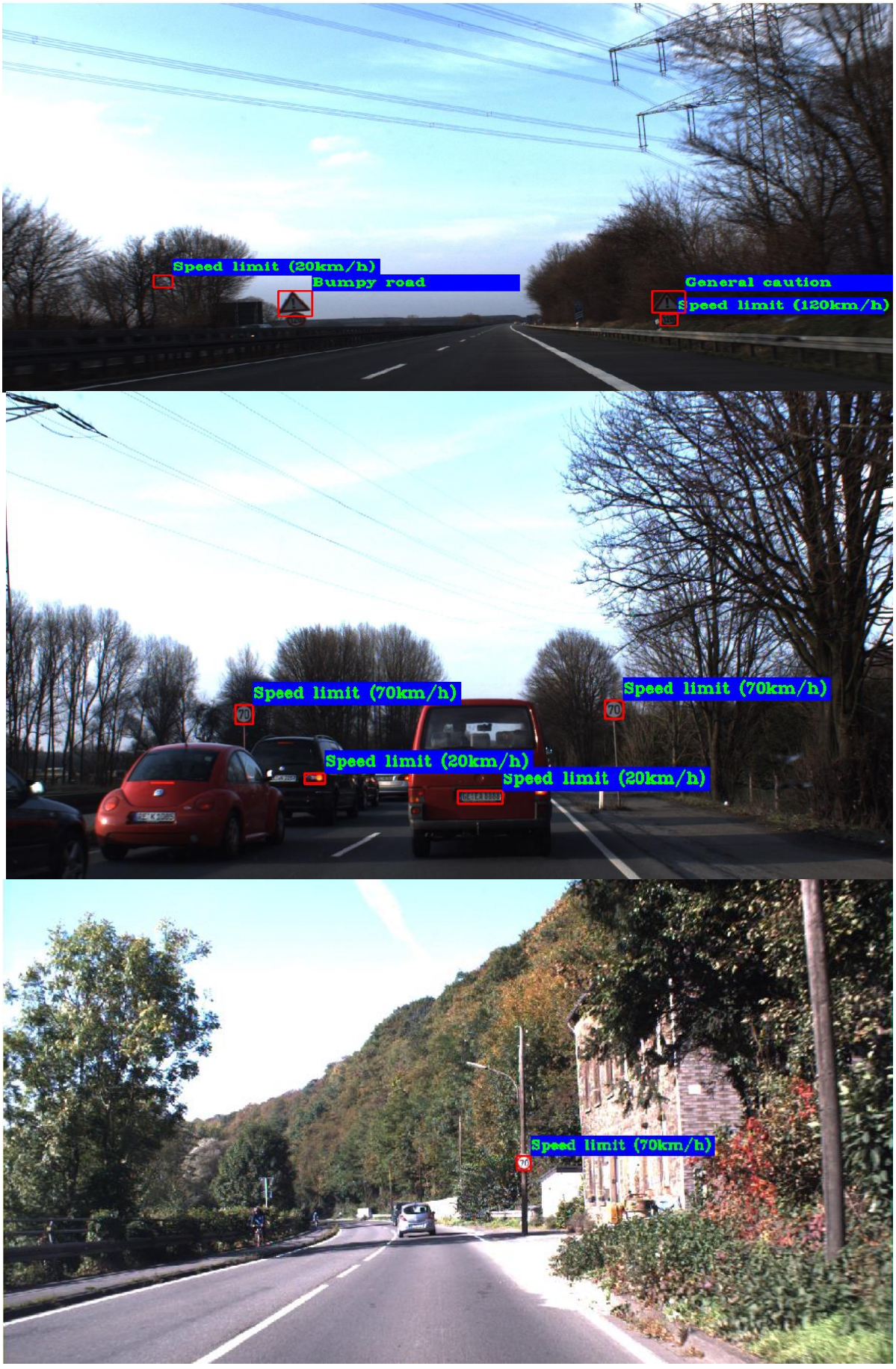

Рис.39- Работа системы (детектирование: Метод границ Кэнни)

5.2 Сравнительный анализ систем

Для оценки качества и сравнения разработанных систем будут рассчитаны основные метрики качества для локализации и классификации (для оценки классификации используются только корректно детектированные знаки, не учитывающие ошибки детектора). Для этого будут проведены 3 эксперимента:

1. Работа системы без внедренного автоэнкодера на тестовых изображениях.
2. Работа системы с импортированным автоэнкодером на тестовых изображениях.
3. Работа системы с видеопотоком (без автоэнкодера в связи сильного торможения программы из-за высокой нагрузки на ЦП).

Результаты для системы, основанной на каскадных классификаторах, приведены в таблицах 7-8

Таблица 7- Результаты работы системы, основанной на каскадных классификаторах, на тестовых изображениях

	Accuracy	Precision	Recall	F1-score
Локализация без автоэнкодера	0.74	0.77	0.62	0.70
Локализация с автоэнкодером	0.67	0.81	0.42	0.56
Точность классификации	0.91	0.89	0.92	0.90

Таблица 8- Результаты работы системы, основанной на каскадных классификаторах, с видеопотоком

	Accuracy	Precision	Recall	F1-score	FPS
Точность локализации	0.9	0.94	0.89	0.91	9

Точность классификации	0.68	0.72	0.70	0.7	___

По результатам из таблиц 7-8 можно сделать следующие выводы:

- Качество детектирования с внедрением автоэнкодера сильно ухудшило работу системы. С его внедрением увеличилась точность, т.к. алгоритм отбрасывает все, что сильно отличается от знаков, уменьшая параметр FP, однако алгоритм так же отбрасывает немалую часть знаков, считая их не похожими на знаки из имеющейся базы, тем самым, сильно увеличивая параметр FN, и, в целом, ухудшая работу системы.
- Точность классификации на тестовых изображениях очень высока, т.к. детектированные знаки из тестовых изображений схожи с обучающей выборкой для сверточной нейронной сети. Однако, по сравнению качеством распознавания тестовых изображений для оценки СНС в главе 4, эта система показывает более плохие показатели. Для Accuracy отличие составляет 0.04. Скорее всего это связано с присутствием примеров, которые необычны для обученной СНС. Это могут быть аффинные искажения (знак частично погнут или находится в непривычном ракурсе) или же цветовые изменения.
- Точность детектирования системы в видеопотоке намного выше, чем на изображениях, т.к. система по мере приближения к знаку может найти его на любом из множества идентичных кадров.
- Точность классификации в видеопотоке ниже, чем на изображениях, т.к. видеопоток взят с авторегистратора российской дороги, где знаки отличаются от немецкой базы дорожных знаков. Также ошибки в классификации можно соотнести с невысоким разрешением видео.

Результаты для системы, основанной на аффинных признаках знаков, приведены в таблицах 9-10

Таблица 9- Результаты работы системы, основанной на аффинных признаках знаков, на тестовых изображениях

	Accuracy	Precision	Recall	F1-score
Без автоэнкодера	0.18	0.11	0.24	0.15
С автоэнкодером	0.36	0.34	0.20	0.25
Точность классификации	0.82	0.82	0.82	0.82

Таблица 10- Результаты работы системы, основанной на афинных признаках знаков, с видеопотоком

	Accuracy	Precision	Recall	F1-score	FPS
Точность локализации	0.12	0.07	0.88	0.14	12
Точность классификации	0.58	0.62	0.59	0.6	___

По результатам из таблиц 9-10 можно сделать следующие выводы:

- С внедрением автоэнкодера в алгоритм, значительно повысилось качество детектирования. Тем не менее, метрики качества сильно ниже допустимого. Данный алгоритм детектирования не приспособлен к задаче распознавания в реальных условиях.
- Точность классификации на тестовых изображениях довольно высокая, однако она хуже показателей СНС в главе 4. Для Accuracy разность составляет 0.13, что является довольно весомым отличием и куда большим, чем для другой системы. Помимо аффинных и цветовых искажений на сильное ухудшение качества классификации может влиять некачественная обрезка знака (например, немного обрезана верхушка знака, что визуально для человека не бросается в глаза, но для СНС затрудняет задачу в разы), и чрезмерная нагрузка на сеть из-за обилия ложных срабатываний детектора данной системы.

- Точность детектирования в видеопотоке, несмотря на высокий показатель полноты, очень низкая, а из-за детектирования и последующей ложной классификации большого количества лишних областей, система перегружается, и начинает проседать частота кадров.

- Точность классификации в видеопотоке ниже, чем на изображениях, т.к. видеопоток взят с авторегистратора российской дороги, где знаки отличаются от немецкой базы дорожных знаков. Также ошибки в классификации можно соотнести с невысоким разрешением видео.

Вывод: Исходя из вышеприведенных таблиц по разработанным системам, можно сделать вывод, что система, построенная на каскадных классификаторах, во всех аспектах показала лучшие показатели качества, чем система, построенная на аффинных признаках знаков, что говорит о высоком уровне качества алгоритмов машинного обучения.

5.3 Выводы по главе

В первом разделе главы были кратко описаны шаги по реализации программных модулей системы распознавания дорожных знаков, а также приведены примеры работы систем.

Во втором разделе главы были рассчитаны и представлены метрики качества систем и сделаны основные выводы по их работоспособности.

ЗАКЛЮЧЕНИЕ

Машинное зрение – сложная и творческая область искусственного интеллекта.

В результате выполнения данного проекта были изучены разные подходы к решению задачи распознавания, а также разработаны и сравнены две системы автоматического распознавания дорожных знаков.

В ходе работы были решены поставленные задачи.

Основные результаты работы:

1. Сформулирована постановка задачи классификации и локализации.
2. Изучены библиотека OpenCV и технологии для детектирования дорожных знаков: детектор границ Кэнни, метод Виолы-Джонса.
3. Построена и обучена модель сверточной нейронной сети для классификации дорожных знаков
4. Реализованы два алгоритма локализации дорожных знаков на изображении с использованием таких подходов, как каскады классификаторов и поиск контуров.
5. Реализована модель сверточного автоэнкодера для борьбы с отсечением ложно-найденных областей на этапе локализации
6. Разработана система, интегрирующая алгоритмы локализации и классификации дорожных знаков.
7. Были проведены эксперименты, которые отобразили точность и эффективность предложенных систем. Также были выявлены проблемы и ограничения. Проведен сравнительный анализ систем на реальных данных. Система, детектор которой основан на алгоритмах машинного обучения сильно обогнала по качеству другую систему, использующую метод поиска контуров.

В целом, данная работа имеет практическое значение для автоматизации процесса распознавания дорожных знаков и может быть использована в областях, связанных с транспортом и безопасностью дорожного движения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. **Желтов С.Ю.,** Обработка и анализ изображений в задачах машинного зрения. // М.: Физматкнига, 2010. — 672 с
2. **Прэтт У.** Цифровая обработка изображений. М.: Мир, 1982. - 311 с.
3. **Николенко, Кадурин, Архангельская**: Глубокое обучение. – СПб.: Питер, 2020. – 480 с.: ил. – (Серия «Библиотека программиста»)
4. **Viola P.A., Jones M.J.** Robust realtime face detection // International journal of computer vision, 2004. Vol. 57, No 2, P. 137154.
5. Open Source Computer Vision Library https://opencv.org/
6. **Xiaoju Ma, Bo Li, Ying Zhang, Ming Yan.** «The Canny Edge Detection and Its Improvement»Kunming University of Science and Technology Kunming, China, 2012
7. Canny Edge Detection https://docs.opencv.org/4.x/da/d22/tutorial_py_canny.html
8. **Lopez, L.** Color-based road sign detection and tracking. Image Analysis and Recognition / L. Lopez, O. Fuentes // Lecture Notes in Computer Science. – Springer. – 2007. – P. 1138-1147
9. **P. Viola and M. Jones**. «Robust real-time object detection» // International Journal of Computer Vision, 2001.
10. База немецких дорожных сцен https://www.kaggle.com/datasets/safabouguezzi/german-traffic-sign-detection-benchmark-gtsdb
11. Contrastlimited adaptive histogram equalization (CLAHE). http://www.mathworks.com/help/images/ref/adapthisteq.html
12. Keras: The Python Deep Learning librar https://keras.io/
13. База дорожных знаков GTSRB https://www.kaggle.com/datasets/meowmeowmeowmeowmeow/gtsrb-german-traffic-sign
14. Распознавание дорожных знаков https://habr.com/ru/articles/554130/
15. **Владимир Паймеров**. Как найти и сравнить похожие изображения автоэнкодером // https://habr.com/ru/articles/703796/