МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
**«Южно-Уральский государственный университет
(национальный исследовательский университет)»**
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

РАБОТА ПРОВЕРЕНА

Рецензент

Ст. преподаватель кафедры
ММОМ ФГБОУ ВО «ЮУрГГПУ»,
к.ф.-м.н.
_____А.М. Шарафутдинова
“___”_____ 2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ

Заведующий кафедрой, д.ф.-м.н.,
профессор
_____ Л.Б. Соколинский
“___”_____ 2019 г.

РАЗРАБОТКА СИСТЕМЫ ОПРЕДЕЛЕНИЯ ПОТЕРИ КОНЦЕНТРАЦИИ ВОДИТЕЛЯ ТРАНСПОРТНОГО СРЕДСТВА С ПРИМЕНЕНИЕМ НЕЙРОСЕТЕВЫХ ТЕХНОЛОГИЙ

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА
ЮУрГУ – 02.03.02.2019.115-105.ВКР

Научный руководитель,
к.ф.-м.н., доцент кафедры СП
_____ С.А. Иванов

Автор работы,
студент группы КЭ-401
_____ В.В. Бессонов

Ученый секретарь
(нормоконтролер)
_____ О.Н. Иванова
“___”_____ 2019 г.

Челябинск–2019

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное автономное образовательное учреждение
высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»
Высшая школа электроники и компьютерных наук
Кафедра системного программирования

УТВЕРЖДАЮ
Зав. кафедрой СП
_____ Л.Б. Соколинский
09.02.2019

ЗАДАНИЕ
на выполнение выпускной квалификационной работы бакалавра
студенту группы КЭ-401
Бессонову Владиславу Витальевичу,
обучающемуся по направлению
02.03.02 «Фундаментальная информатика и информационные технологии»

1. Тема работы (утверждена приказом ректора от «25» апреля 2019 № 899)

Разработка системы определения потери концентрации водителя транспортного средства с применением нейросетевых технологий.

2. Срок сдачи студентом законченной работы: 05.06.2019.

3. Исходные данные к работе

3.1. Begino Y., Goodfellow I., Courville A. Deep Learning. – MIT Press, 2016. – 773 p.
3.2. Хайкин С. Нейронные сети: полный курс. // Под ред. Н.Н. Куссуль, 2-е изд. – ООО «И.Д. Вильямс», 2016. – 1104 с.

4. Перечень подлежащих разработке вопросов

4.1. Провести анализ предметной области и аналогичных решений.
4.2. Разработать алгоритмы распознавания потери концентрации на изображении.
4.3. Спроектировать топологию нейронной сети.
4.4. Осуществить сбор и предобработку данных для обучения.
4.5. Реализовать и обучить нейронную сеть.
4.6. Реализовать и протестировать приложение для распознавания признаков утомления и потери концентрации у человека на видео.

5. Дата выдачи задания: 08.02.2019.

Научный руководитель

к.ф.-м.н., доцент кафедры СП С.А. Иванов

Задание принял к исполнению В.В. Бессонов

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

ОСНОВНЫЕ ОПРЕДЕЛЕНИЯ

Искусственная нейронная сеть (ИНС) – математическая модель, построенная по принципу организации и функционирования биологических нейронных сетей нервных клеток живого организма [19].

Машинное обучение – класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решению множества сходных задач [1].

Глубокое обучение – часть методов машинного обучения, которые моделируют высокоуровневые абстракции в наборах данных, используя архитектуры, состоящие из множества нелинейных преобразований.

Обучающая выборка – выборка, по которой производится настройка нейронной сети.

Контрольная (тестовая) выборка – выборка, по которой оценивается качество построенной сети.

Классификация – один из разделов машинного обучения, посвященный решению следующей задачи. Имеется множество объектов (ситуаций), разделенных некоторым образом на классы. Задано конечное множество объектов, для которых известно, к каким классам они относятся. Это множество называется обучающей выборкой. Классовая принадлежность остальных объектов не известна. Требуется построить алгоритм, способный классифицировать произвольный объект из исходного множества [14].

АКТУАЛЬНОСТЬ ТЕМЫ

Автомобиль является не только неотъемлемой частью жизни людей, но и одним из наиболее опасных транспортных средств. Ежегодно миллионы людей погибают в автомобильных авариях. Согласно официальным данным, в мире ежегодно, в результате ДТП гибнет около 1,2 миллиона человек, или 3287 человек в день [16]. Наиболее опасными являются трассы, так как ограничения по скорости на них не такие жесткие, как в населенных пунктах. Огромные скорости нередко граничат со смертельными

случаями. Немалую роль в этом играет человеческий фактор. Одной из часто встречаемых причин смертности в результате ДТП, является сонливость водителя во время управления автомобилем и это причина каждого пятого дорожного происшествия. Поэтому наиболее опасным на трассе по праву считается переутомление, которое ведет к потере концентрации и неспособностью управлять транспортным средством. Долгая прямая дорога без светофоров и пешеходных переходов усыпляет бдительность водителя, а вместе с ней и его самого. Такому пагубному воздействию со стороны дороги в большей степени подвержены дальнобойщики. Их ошибки, связанные с усталостью могут стоить человеческих жизней. Так, например, 6 апреля 2017 года в Рязанской области лоб в лоб столкнулись грузовик и рейсовый автобус. В аварии погибли два человека, более тридцати получили ранения. В числе пострадавших оказались восемь детей. ДТП произошло по вине водителя грузовика, который заснул за рулем [18]. И это далеко не единственный случай ДТП со смертельным исходом по вине заснувшего водителя.

Обеспечение безопасности жизнедеятельности людей является одной из наиболее приоритетных задач в мире на сегодняшний день. Ведущие специалисты в области автомобилестроения оснащают свои автомобили передовыми технологиями, уменьшающими влияние человеческого фактора при вождении. Всевозможные датчики выполняют роли органов чувств человека и односмысленно трактуют различные ситуации на дороге, выбирая (в большинстве случаев) рациональную стратегию поведения в экстренной ситуации. Однако работа упомянутых выше технологий еще не совершенна и требует доработки. Исследования в этой области активно ведутся, и плоды данной работы можно ежегодно наблюдать на очередной презентации автомобилей-флагманов мировых брендов.

ЦЕЛЬ И ЗАДАЧИ ИССЛЕДОВАНИЯ

Целью данной работы является разработка системы распознавания признаков усталости и потери концентрации у водителя во время управле-

ния транспортным средством в режиме реального времени на основании видео. Система берет на вход видеофайл, разбивает его на кадры и обрабатывает каждый отдельный кадр. Определяет у водителя закрытость глаз, направления взгляда водителя, зевание и положение головы. При продолжительном положительном результате (несколько кадров) выдает общий положительный ответ об отвлеченности водителя во время управления транспортным средством. Для выполнения этой цели необходимо решить подзадачи, указанные ниже.

1. Провести анализ предметной области и аналогичных решений.
2. Разработать алгоритмы для распознавания потери концентрации на изображении.
3. Спроектировать топологию нейронной сети.
4. Осуществить сбор и предобработку данных для обучения.
5. Реализовать и обучить нейронную сеть.
6. Реализовать и протестировать приложение для распознавания признаков утомления и потери концентрации у человека на видео.

СТРУКТУРА И ОБЪЕМ РАБОТЫ

Работа состоит из введения, пяти глав и заключения. Объем работы составляет 45 страниц, объем списка литературы – 20 источников.

СОДЕРЖАНИЕ РАБОТЫ

В первой главе «Анализ предметной области» рассматриваются исследования по использованию нейросетевых технологий в области, в рамках которой выполняется данная работа и обзор уже существующих решений поставленной задачи.

Вторая глава «Теоретическая часть» содержит подробное описание алгоритмов, применяемых в решении поставленной задачи.

Третья глава «Проектирование» содержит общее описание архитектуры системы, описание этой системы компонентов и ее реализацию.

В четвертой главе «Реализация системы» приводится техническая реализация системы на основе поставленного списка требований, а также

описание подобранной топологии нейронной сети.

В пятой главе «Тестирование» приведены результаты тестирования.

В заключении приводятся основные результаты выполненной работы.

Приложения содержат дополнительные материалы работы: листинги реализованных модулей системы.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Обзор аналогичных проектов

Данная работа реализована с использованием сверточных нейронных сетей. Разработкой всевозможного программного обеспечения для помощи водителю при вождении, как говорилось во введении, занимаются ведущие автопроизводители.

Mercedes-Benz Attention Assist [5]

Одним из наиболее ярких примеров можно по праву считать систему Mercedes-Benz Attention Assist. Данная система осуществляет наблюдение за поведением водителя автомобиля, анализируя следующие данные: действие водителя во время езды, управление рулем, способ управления машиной. Схематически система состоит из датчика руля, контрольной лампочки и звукового предупреждения водителя. Датчик руля определяет усилие, оказываемое на руль при его вращении и его изменение. Помимо этого, система учитывает показания других контрольных датчиков автомобиля: тормозной системы, устойчивости при движении, параметров двигателя и ограничения видимости.

Во время подачи сигналов на прибор управления обрабатываются и определяются следующие параметры:

1) анализируется скорость и боковое ускорение автомобиля в первые 30 минут с начала движения. Это необходимо для определения стиля вождения;

2) определяется условие, при котором происходит движение: длительность поездки и время суток;

3) анализируется управления некоторыми отдельными частями автомобиля: тормозной системы, кнопок, находящихся на щитке управления, а также переключателями, расположенными под рулем;

4) определяется сила воздействия на рулевое колесо;

5) анализируется состояние дорожного покрытия и поведение автомобиля во время движения (боковое и продольное ускорение).

Итог определяется методом установки нарушений в действиях водителя и изменения направления движения автомобиля. На экран приборной панели посылается сигнал в звуковом сопровождении о требовании остановиться для отдыха. В случае игнорирования предупреждения, когда водитель в сонном состоянии не прекращает движение, система продолжает сигнализировать через каждые 15 минут.

Volvo Driver Alert Control [3]

Аналогично компании Mercedes-Benz, работы по контролю за внимательностью за рулем при помощи компьютеризированных систем осуществляет компания Volvo. Система Driver Alert Control от Volvo отличается от контроля Attention Assist тем, что определяет только траекторию движения авто по дороге, а видеоконтроль, установленный в направлении пути следования автомобиля, определяет его нахождение на полосе дорожного покрытия. Если происходит уход от установленных границ, система реагирует на это как на признаки усталости водителя. Выдается два вида предупреждающих сигналов: «жесткий» и «мягкий», которые зависят от общего самочувствия водителя. Сигналы отличаются друг от друга тональностью и громкостью звука.

Seeing Machines by General Motors [7]

Оценку утомляемости взгляда водителя дает контролирующая схема, которую устанавливает компания General Motors, и где базой является испытанная методика Seeing Machines, используемая как на грузовом транспорте, так и железнодорожном, а также при карьерных разработках. Специально встроенный блок производит контроль открытости глаз водителя и их сосредоточенности. Обнаружив признаки усталости, состояние близкого ко сну и потерю внимательности водителем, система дает команду о прекращении движения транспортного средства.

Вывод из анализа существующих аналогов

В ходе анализа была сформулирована следующая задача: выявить очевидные признаки усталости водителя. Данную задачу можно разбить на

подзадачи: определение на фотографии лица человека и его положения по отношению к камере, определение на лице глаз, рта, носа, контура лица, анализ потери концентрации на основе выделенных выше объектов лица.

1.2. Обзор существующих решений для создания нейронных сетей

TensorFlow [12] – открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия [9].

Keras [10] – открытая нейросетевая библиотека, написанная на языке Python. Она представляет собой надстройку над фреймворками TensorFlow и Theano. Нацелена на оперативную работу с сетями глубинного обучения, при этом спроектирована так, чтобы быть компактной, модульной и расширяемой [4].

OpenCV [11] – библиотека алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения с открытым кодом [6].

Dlib [17] – кроссплатформенная программная библиотека, содержащая алгоритмы машинного обучения и инструменты для создания сложного программного обеспечения. Используется как в промышленности, так и в академических кругах в широком диапазоне областей, включая робототехнику, встроенные устройства, мобильные телефоны и большие высокопроизводительные вычислительные среды.

Выводы по первой главе

В первой главе был проведен обзор аналогов. Наличие аналогов среди популярных брендов в сфере автомобилестроения говорит об актуальности проблемы. В соответствии с целями работы в первой главе был проведен обзор аналогов и уже имеющихся решений для упрощения реализации проекта. Наличие аналогов, реализованных по разным технологиям,

подтверждает актуальность поставленной задачи, а большой выбор программных библиотек, предназначенных для работы с нейронными сетями, показывает их популярность.

По итогу главы были выбраны вспомогательные средства для реализации системы и требования к которой сформулированы во второй главе.

2. ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

2.1. Обнаружение на изображении лица человека

Одна из основных задач выявления признаков потери концентрации у человека на изображении состоит в определении расположения лица. Для данной цели подходят несколько общедоступных библиотек. В данной работе я буду использовать общедоступную библиотеку dlib. Данная библиотека предоставляет возможности поиска лиц на изображении с использованием алгоритма HOG (Histogram of Oriented Gradients).

Основная идея алгоритма HOG заключается в предположении, что внешний вид и форма лица на участке изображения могут быть описаны распределением градиентов интенсивности [8].

Данный алгоритм работает следующем образом. Исходное изображение преобразуется в градации серого. Далее для каждого пикселя изображения определяется градиент в контексте соседних пикселей.

На следующем шаге изображение разбивается на небольшие квадраты 16x16 пикселей называемые ячейками. Для каждой ячейки вычисляется гистограмма направлений градиентов, с помощью участия каждого пикселя ячейки во взвешенном голосовании для девяти каналов гистограммы направлений.

Затем ячейки группируются в более крупные квадратные блоки – дескрипторы, где происходит нормировка гистограмм ячеек. Каждая ячейка может входит в более чем один дескриптор. Конечным шагом является подача дескриптора на предобученный SVM-классификатор [2]. На рис. 1 представлен пример работы алгоритма HOG.

На основе алгоритма HOG работает заранее обученная модель нейронной сети shape_predictor_68_face_landmarks.dat, доступная для использования при помощи библиотеки dlib. Работа данной модели заключается в поиске на зарнее выделенном изображении человеческого лица и 68 ключевых точек на нем. Пример работы данной модели представлен на рис. 2.

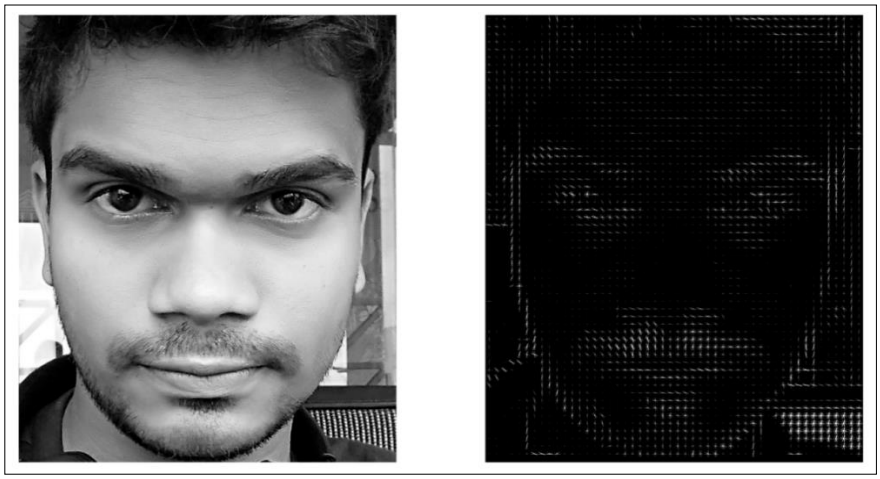

Рис. 1. Пример работы алгоритма HOG

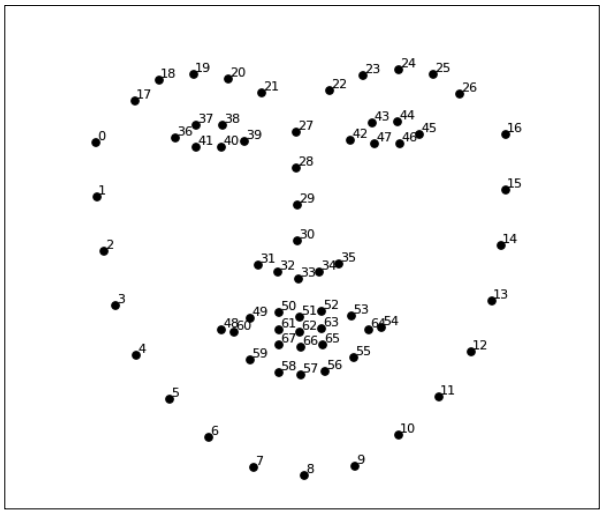

Рис. 2. Пример работы модели нейронной сети shape_predictor_68_face_landmarks.dat

2.2. Сверточные нейронные сети

Наиболее приемлемым решением для задачи эффективного распознавания изображений являются сверточные нейронные сети (англ. Convolutional Neural Network, CNN) (в определения) – специальная архитектура искусственных нейронных сетей, помогающая уменьшить вектор признаков и заключающая в себе 3 основные парадигмы: локальное восприятие, разделяемые веса и субдискретизация. Проблемы объема памяти, производительности и времени обучения всегда актуальны при использовании нейронных сетей. Пример топологии сети приведен на рис. 3.

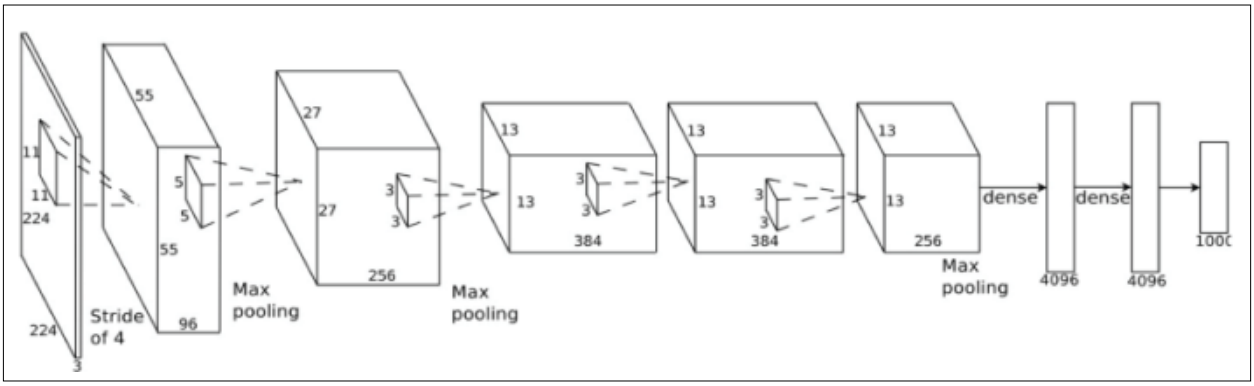

Рис. 3. Пример топологии сверточной нейронной сети

Локальное восприятие подразумевает, что на вход одного нейрона подается не все изображение, а лишь некоторая его область. Такой подход позволил сохранять топологию изображения от слоя к слою.

Разделяемые веса предполагают, что для большого количества связей используется очень небольшой набор весов. Например, при наличии на входе изображения 28x28 пикселя каждый из нейронов следующего слоя примет на вход только небольшой участок этого изображения размером 3x3, причем каждый из фрагментов будет обработан одним и тем же набором весов. Такое введенное ограничение на веса только улучшает обобщающие свойства сети, что в итоге позитивно сказывается на работоспособности сети.

Суть субдискретизации заключается в уменьшении пространственной размерности изображения. Изображение грубо уменьшается в заданное количество раз. Субдискретизация нужна для обеспечения инвариантности к масштабу.

Предварительная обработка изображений, полученных в результате разбиения видеофайла на кадры и правильный выбор архитектуры нейронной сети необходимы для достижения высокой точности распознавания системы и улучшения ее производительности. В качестве входных данных для нейронной сети будут использоваться видеофайлы любого размера в формате .avi, .mp4, .wmv или .mpeg. Поскольку в работе будет использоваться сверточная нейронная сеть, которая благодаря использова-

нию концепции разделяемых весов имеет способность реагировать главным образом на изображения, то в процессе предобработки изображений процедура избавления от шумов использоваться не будет.

2.3. Алгоритмы для определения признаков потери концентрации

Для повышения точности работы приложения помимо нейронной сети, самостоятельно построенной, обученной и используемой в приложении было принято решение разработать набор алгоритмов распознавания признаков потери концентрации у человека на изображении с использованием в качестве входных данных результат работы нейронной сети shape_predictor_68_face_landmarks.dat, а именно 68 ключевых точек на лице человека. Каждый из алгоритмов отвечает за отдельный признак потери концентрации.

Из всех признаков потери концентрации были выделены основные четыре: закрытость глаз, отвлеченный от дороги взгляд водителя, положение головы не по направлению движения и зевание как сигнализатор утомленности и сонливости у водителя.

Закрытость глаз

Главным критерием того, что глаза у человека на изображении закрыты, является сомкнутость век. Из 68 ключевых точек за границы глаз (веки) отвечают следующие точки: 36, 37, 38, 39, 40, 41 – за правый глаз и 42, 43, 44, 45, 46, 47 – за правый глаз. Ключевые точки для анализа закрытости глаз у человека на изображении показаны на рис. 4.

Для определения закрытости глаза нам необходимо построить два пересекающихся отрезка: горизонтальный – с концами на внутреннем и наружным уголках глаза (ось глазной щели), соответствующими 36 (42) и 39 (45) ключевым точкам, и вертикальный – с концами на серединах верхнего и нижнего века, соответствующими серединами между 37 и 38 (43 и 44) и 40 и 41 (46 и 47).

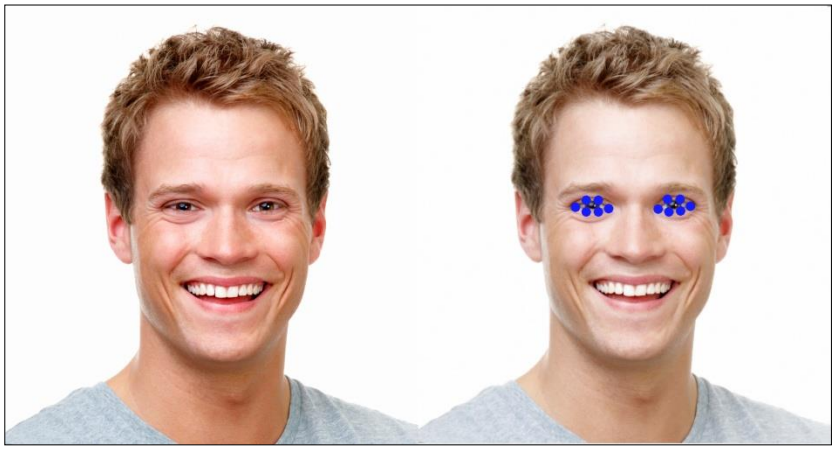

Рис. 4. Ключевые точки для анализа закрытости глаз у человека

Используя длины полученных отрезков мы можем получить их отношение. Данная величина является наиболее точным критерием определения закрытости или приоткрытости глаз.

Отвлеченный от дороги взгляд водителя

Направление взгляда человека характеризуется положением зрачка в видимой части глазного яблока. В случае, если зрачок расположен не в середине видимой части глазного яблока, мы можем утверждать, что взгляд человека направлен не прямо перед собой, а в сторону. В таком случае мы можем предположить, что человек не сконцентрирован на дороге во время управления транспортным средством.

Главной задачей определения положения зрачка внутри видимой части глазного яблока является выделение зрачка на фоне глазного яблока. Основным отличительным критерием зрачка от глазного яблока является оттенок: в случае с глазным яблоком – светлый; в случае со зрачком – темный.

Разделим изображение глаза на две зоны – правую и левую. В том случае, если зрачок расположен по центру, отношение светлой части глаза к темной приблизительно одинаковое на обеих зонах. Если в одной зоне отношение светлой части к темной значительно превосходит данное отношение в другой зоне, мы можем считать, что взгляд направлен не вперед, что, в свою очередь говорит об отвлеченности водителя во время управле-

ния транспортным средством. Пример предобработанного изображения глаза для анализа отвлеченного от дороги взгляда водителя представлен на рис. 5.

Рис. 5. Пример предобработанного изображения глаза для анализа отвлеченного от дороги взгляда водителя

Положение головы не по направлению движения

В случае, когда голова водителя не ориентирована в сторону направления движения, можно предположить, что обзор водителя недостаточен для безопасного управления транспортным средством.

Для определения положения головы нам удобнее всего воспользоваться ключевыми точками 0, 16 и 30. Данные точки отвечают за левый и правый виски и кончик носа. В ситуации, когда голова водителя направлена вперед, мы видим лицо а анфас и нос расположен посередине. Если же расстояние от кончика носа до какого либо из висков значительно больше чем расстояние до другого виска, то голова водителя направлена в сторону. Ключевые точки для анализа положения головы у человека на изображении показаны на рис. 6.

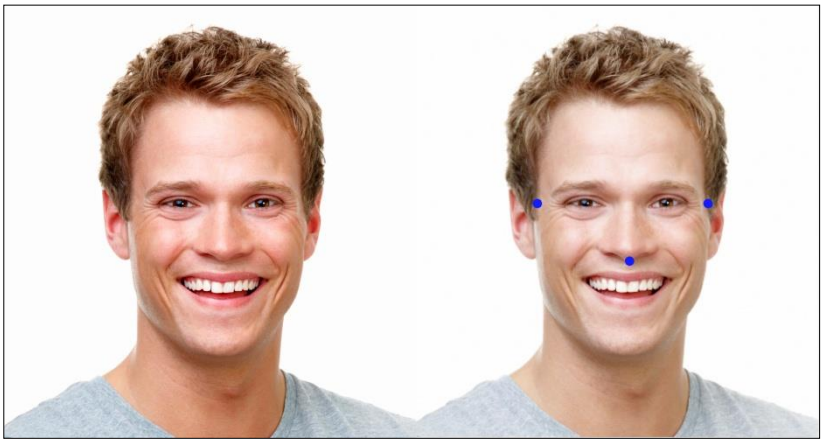

Рис. 6. Ключевые точки для анализа положения головы у человека на изображении

Зевание

В отличие от рассмотренных выше признаков потери концентрации у человека, зевание не является прямым признаком потери концентрации водителя, однако оно является сигнализатором утомленности и сонливости, которые, в свою очередь, ведут к потере концентрации. В связи с этим, диагностирование потери концентрации также играет важную роль.

При зевании человек широко раскрывает рот, что, в свою очередь, можно считать основным показателем зевания человека. Широко раскрытый рот характеризуется увеличенным расстояниям между серединами верхней и нижней губ. За губы человека отвечают ключевые точки с 45 по 67. Однако нам потребуются только шесть центральных точек верхней губы – 50, 51, 52, 53, 62, 63 и шесть центральных точек нижней губы – 56, 57, 58, 59, 66, 67. Для того, чтобы толщина губ не влияла на работу алгоритма нам необходимо брать координату точки, равноудаленой от шести центральных точек верхней губы и координату точки, равноудаленой от шести центральных точек нижней губы. Две данные точки примем за середины верхней и нижней губ. Расстояние между этими точками характеризует степень открытости рта, по которой, в свою очередь, мы сможем судить о том зевает человек на изображении в данный момент или нет. Ключевые точки для анализа зевания у человека на изображении показаны на рис. 7.

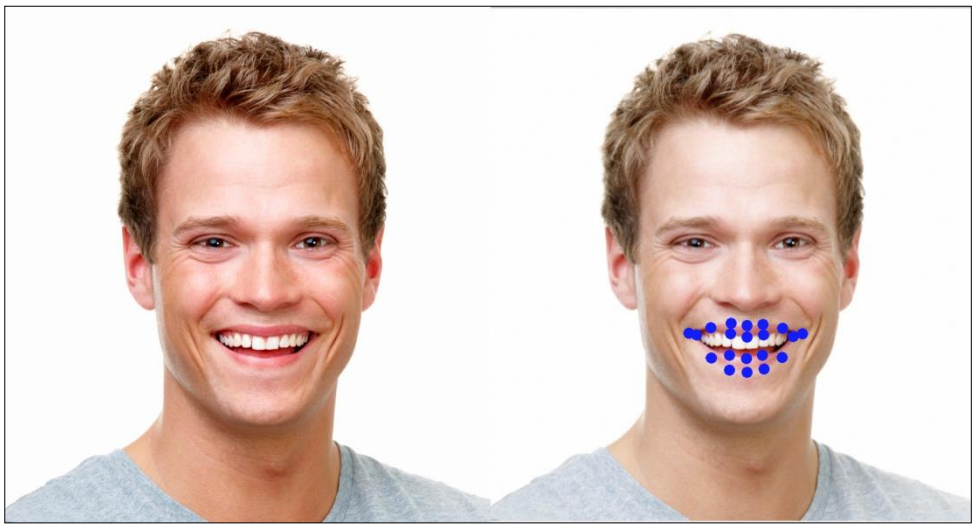

Рис. 7. Ключевые точки для анализа зевания у человека на изображении

Выводы по второй главе

Во второй главе были рассмотрены теория сверточных нейронных сетей, принципы работы сверточных нейронных сетей. Были разработаны алгоритмы распознавания признаков потери концентрации у водителя во время управления транспортным средством на основе ключевых точек, являющихся результатом работы рассмотренной модели нейронной сети.

3. ПРОЕКТИРОВАНИЕ

3.1. Функциональные требования

Функциональные требования описывают поведение системы, т.е. ее действия (вычисления, преобразования, проверки, обработку и т.д.) [15].

Разрабатываемая система должна удовлетворять функциональным требованиям, приведенным ниже.

1. Система должна предоставлять возможность выбора заранее записанного видеофайла в качестве входных данных для анализа.
2. Система должна иметь возможность анализировать видеопоток, считываемый с веб-камеры в режиме реального времени.
3. Система должна классифицировать признаки утомления у водителя, изображенного на видео.
4. Система должна вывести на экран сообщение с предупреждением в случае обнаружения признаков потери концентрации.

3.2. Нефункциональные требования

Нефункциональные требования описывают свойства системы (удобство использования, безопасность, надёжность, расширяемость и т.д.), которыми она должна обладать при реализации своего поведения [15].

В ходе обзора уже имеющихся решений для реализации приложения, представленного в первой главе, были найдены необходимые библиотеки, поэтому разрабатываемая система должна удовлетворять нефункциональным требованиям, приведенным ниже.

1. Система должна быть реализована на языке Python.
2. Система должна использовать открытую программную библиотеку для создания нейронных сетей Keras.
3. Система должна использовать открытую программную библиотеку алгоритмов компьютерного зрения OpenCV.
4. Система должна использовать открытую программную библиотеку машинного обучения dlib.

5. Система должна быть единой и не требовать от пользователя дополнительных действий для работы, кроме запуска и указания входных данных и параметров.

3.3. Топология нейронной сети

Для решения поставленной задачи была разработана следующая топология нейронной сети, представленная на рис. 8.

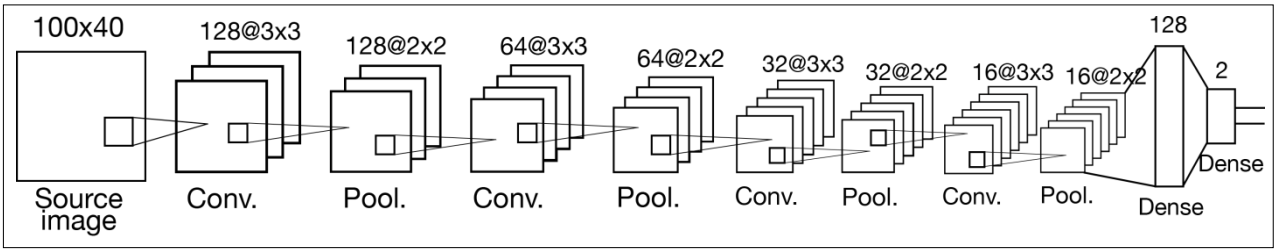

Рис. 8. Топология нейронной сети

Даная топология нейронной сети была подобрана эмпирическим путем. Она содержит 4 сверточных слоя, 4 субдискретизирующих слоя и 2 полносвязных слоя.

Задача сверточных слоев заключается в получении признанного описания изображения в виде набора карт признаков.

Задачей субдискретизирующих слоев является уменьшение размерности при помощи метода выбора максимального элемента (max-pooling) – вся карта признаков разделяется на ячейки, из которых выбираются максимальные по значению

В полносвязных слоях каждый нейрон соединен со всеми нейронами на предыдущем уровне, причем каждая связь имеет свой весовой коэффициент.

Входными данными у нейронной сети будет картинка размером 100x40 пикселей, представленная в цифровом виде (двумерный массив пикселей). Выход нейронной сети представляет собой число двоичной системы счисления, где 0 – водитель не подает признаков усталости, 1 – водитель подает признаки усталости.

3.4. Предварительная обработка входных данных

С целью повышения точности работы нейронной сети, используемой в алгоритме определения потери концентрации у водителя, необходимо провести предобработку исходных данных, переведя их в требуемый для работы нейронной сети формат и исключив из них всю ненужную информацию.

В качестве входных данных используется видеопоток, полученный либо со встроенного видеозаписывающего устройства (веб-камеры) в режиме реального времени, либо записанный заранее видеофайл.

Первый этап предобработки заключается в переводе входного видеопотока в последовательность изображений. Для получения видеофайла и его последующего разбиения на кадры используется библиотека OpenCV.

Второй этап предобработки заключается в определении на изображении лица человека.

Третьим этапом является выделение области глаз на ранее выделенном участке изображения с лицом человека и изменение размера данной области в соответствии с требованиями к входным данным нейронной сети.

Четвертым этапом предобработки является изменение размеров ранее выделенного изображения области глаз.

Результатом предобработки является изображение области глаз размером 100x40 пикселей.

3.5. Проектирование работы системы распознавания потери концентрации у человека

Для определения признаков потери концентрации у водителя было решено использовать самостоятельно обученную модель сверточной нейронной сети, решающую задачу классификации изображений с лицами людей с закрытыми и открытыми глазами, совместно с алгоритмами опре-

деления потери концентрации у человека с использованием заранее обученной модели нейронной сети shape_predictor_68_face_landmarks.dat.

Главная функция системы начинает свое выполнение с отрисовки окна интерфейса. После выбора пользователем источника данных, в зависимости от того, выбран заранее записанный видеофайл или запись ведется в режиме реального времени, проводится либо считывание файла из указанной директории, либо начинается запись видеопотока с веб-камеры. Далее происходит разбиение видео на кадры, после чего вызывается функция предобработки, в результате которой мы получаем изображение области глаз размером 100x40 пикселей. Необработанный кадр анализируется функциями потери концентрации на основе модели заранее обученной модели нейронной сети shape_predictor_68_face_landmarks.dat. Предобработанный кадр анализируется самостоятельно реализованной и обученной нейронной сетью. Таким способом обрабатываются все кадры видеофайла. В случае, когда хотя бы одна из функций возвращает значение True (кроме функции распознавания зевания) на протяжении трех секунд, пользователю выводится сообщение о том, что обнаружены признаки потери концентрации. Если функция распознавания зевания выдает положительное значение хотя бы три раза на промежутке времени в одну минуту, то пользователю выводится сообщение о том, что обнаружены признаки утомления и сонливости.

3.6. Варианты использования системы

Для проектирования был использован язык UML. Была построена модель взаимодействия внешнего актера с системой распознавания в виде диаграммы вариантов использования (use-case diagram) [20]. Полученная диаграмма изображена на рис. 9.

Актеры, взаимодействующие с системой:

Пользователь - человек, использующий систему распознавая признаков утомления и потери концентрации и ее определенные функции.

Краткое описание вариантов использования:

Пользователю доступны следующие функции.

1. Запустить распознавание.

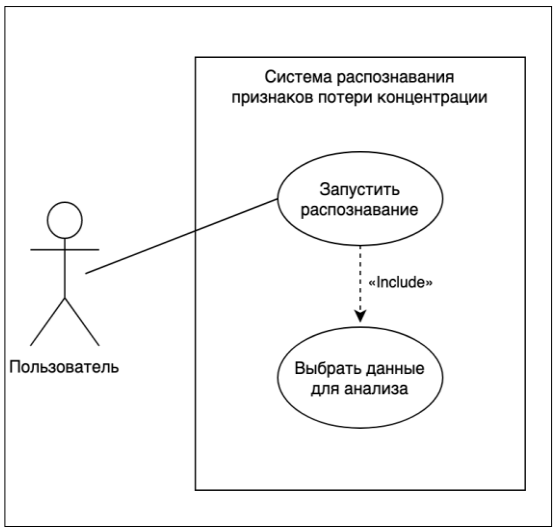

Рис. 9. Диаграмма вариантов использования

3.7. Проектирование архитектуры системы

Диаграмма деятельности – технология, позволяющая описывать логику процедур, бизнес-процессы и потоки работ [20].

На рис. 10 представлена диаграмма деятельности системы распознавания признаков усталости.

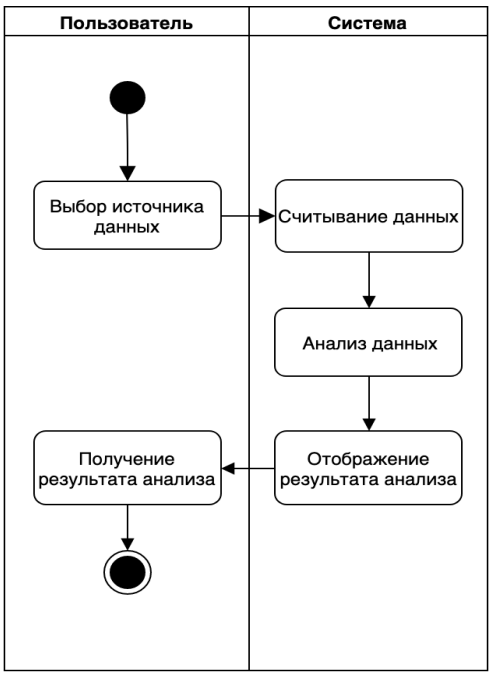

Рис. 10. Диаграмма деятельности системы

При загрузке изображения пользователем в программу начинается последовательная предобработка изображения, заключающаяся в поиске на изображении области лица и глаз. Если лицо не будет обнаружено, будет выведено сообщение об ошибке в поле результата программы.

При успешной предобработке автоматически запустится алгоритм распознавания потери концентрации у водителя по изображению. Нейронная сеть вынесет вердикт, который будет отображен в поле результата программы.

Выводы по третьей главе

В третьей главе с учетом назначения системы и обзора готовых решений для ее реализации, проведенного в первой главе, были определены и сформулированы функциональные и нефункциональные требования, представлена диаграмма вариантов использования системы и спецификации прецедентов. Также спроектирована система для распознавания признаков усталости и потери концентрации у водителя.

Реализация системы представлена в четвертой главе.

4. РЕАЛИЗАЦИЯ

СРЕДСТВА РАЗРАБОТКИ

Для разработки программной части для распознавания признаков потери концентрации на видео был использован высокоуровневый язык программирования Python 3.7. Разработка системы велась в среде разработки PyCharm Community 2017.1 и на операционной системе Mac OS Mojave 10.14.4. Для языка программирования Python было использованы стандартные библиотеки Python, а также Keras и т.д. (Расписать библиотеки)

ФОРМИРОВАНИЕ ОБУЧАЮЩЕЙ ВЫБОРКИ

Формирование множества обучающих данных имеет принципиально важное значение для успешного решения задач машинного обучения. Часто задачи машинного обучения сводятся именно к правильному формированию обучающего множества. Из-за активного развития нейронных сетей проблема формирования обучающей выборки очень актуальна, поскольку во многих задачах глубокие нейронные сети демонстрируют качество, существенно превосходящее остальные алгоритмы машинного обучения. В то же время, в современной литературе по машинному обучению вопросы формирования обучающего множества полностью игнорируются, или им уделяется недостаточное внимание [13].

В качестве обучающих данных был использован датасет «Closed Eyes In The Wild (CEW)». Данный датасет состоит из 2500 фотографий с изображенными на них лицами людей с закрытыми и открытыми глазами.

Размер готовой обучающей и контрольной выборки составил 2500 изображений, разделенных на 2 разных класса. В одном классе – изображения людей с открытыми глазами, в другом – с закрытыми.

4.1. Реализация нейронной сети

Для выполнения поставленной задачи, заключающейся в классификации изображений людей с закрытыми и открытыми глазами, была реализована нейронная сеть на основе топологии, выбранной в главе «Проектирование».

Для создания и обучения модели сверточной нейронной сети была выбрана библиотека Keras. Данная библиотека обладает удобными для инструментами для создания модели нейронной сети согласно заранее выбранной топологии, и ее дальнейшего обучения.

Обучение модели нейронной сети производилось в 147 эпох на персональном компьютере с видеокартой nVidia GeForce GTX 1050. Время обучения составило 1 час 45 минут. Точность классификации, рассчитываемая как отношение количества изображений, по которым классификатор принял правильное решение и общего количества изображений в выборке, на тестовой выборке оказалась равна 0.89 или 89 %.

4.2. Программная реализация алгоритмов для распознавания потери признаков концентрации

Для выявления признаков утомления и потери концентрации у водителя, помимо самостоятельно обученной и протестированной нейронной сети, было решено воспользоваться заранее обученной моделью нейронной сети shape_predictor_68_face_landmarks.dat. С помощью данной модели и библиотеки dlib можно размечать лицо человека на изображении 68-ю ключевыми точками. Алгоритмы работы с данными точками для распознавания признаков потери концентрации и утомления у человека на изображении приведены в главе «Теоретическая часть».

Для удобства при вычислительной работе с координатами ключевых точек была использована библиотека math. Для определения положения зрачка использовались функции библиотеки OpenCV.

Каждый из описанных в главе «Теоретическая часть» алгоритмов был программно реализован как отдельная функция. Каждая функция принимает в качестве входных данных координаты ключевых точек, необходимых для проведения вычислений в данной функции и возвращает в качестве результата True в случае, если признаки потери концентрации были выявлены, и False, если признаки потери концентрации выявлены не были.

Главная функция системы, отвечающая за получение данных, их обработку и выдачу результата, использует все реализованные функции распознавания признаков потери концентрации у человека и обрабатывает результаты работы данных функций.

4.3. Реализация системы распознавания потери концентрации

Задача системы распознавания потери концентрации заключается считывании данных для анализа, обработке полученных данных, анализа данных с целью выявления признаков потери концентрации и отображения результата анализа при положительном вердикте.

Программная реализация была осуществлена с использованием библиотек OpenCV для корректного считывания входного видеопотока и его последующего разбиения на кадры, предобработки данных для работы нейронной сети и отображения результатов работы системы; dlib для распознавания на изображении лица человека с выделением ключевых точек для дальнейшей обработки алгоритмами распознавания признаков утомления и потери концентрации; math для удобной работы с вычислениями в алгоритмах.

Главная функция программы, ответственная за отрисовывание интерфейса, считывание видеофайла и его дальнейшие обработку и анализ с отображением результата пользователю, содержит две основных функции, помимо отрисовывания интереса. Данные функции отвечают за корректное считывание видеопотока и аналитическую обработку считанных файлов.

При запуске программы пользователю предоставляются на выбор два варианта использования программы. Первый заключается в выборе заранее записанного видеофайла для анализа. Для этого пользователю необходимо нажать на кнопку «Выбрать файл». Далее пользователю нужно выбрать файл в открывшемся проводнике и нажать на кнопку «open». После этого начнется выполнение функции считывания видеофайла, его последующее разбиение на кадры и последовательную передачу кадров блок

аналитической обработки. За выбор файла и его последующее считывание отвечают библиотеки PyQt5 и OpenCV соответственно. Листинг выбора и считывания файла, а также разбиения на кадры приведен на рис. 11.

```
def select_file(self):
        is_yawning = False
        yawning_true = False
        y_times = []
        time_s = time.time()
        filename = str(QFileDialog.getOpenFileName(self, 'Open file',
                                                   'test/', "*.avi *.mp4 *.wmv
*.mpeg"))
        video = cv2.VideoCapture(filename)
        while True:
            frame = video.read()[1]
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = detector(gray)
            for face in faces:
              x, y = face.left(),face.top()
              width, height = face.right() - x, face.bottom() - y
                landmarks = predictor(gray, face)
                if yawning_true:
                    cv2.putText(frame, 'Asleep', (650, 100),
cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 3)
                elif yawning_detect(list_of_points[3], landmarks):
                    if not is_yawning:
                        y_times.append(time.time())
                        is_yawning = True
                        if (len(y_times) >= 3) and (y_times[-1] - y_times[-
3] <= 60):
                            yawning_true = True
                else:
                    is_yawning = False
                if blinking_detect(list_of_points[0], list_of_points[1],
landmarks) or \
                        glance_detect(list_of_points[0], list_of_points[1],
landmarks, gray) or \
                        head_detect(list_of_points[2], landmarks) or \
                      blinking_recognition(frame, x, y, width, height):
                    if time.time() - time_s >= 3:
                        cv2.putText(frame, 'ALARM!', (50, 100),
cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 3)
                else:
                    time_s = time.time()
            cv2.imshow("Detector", frame)
            cv2.waitKey(1)
```

Рис. 11. Листинг выбора файла, считывания файла и вызова функции аналитической обработки

Вторая функция выполняет задачу считывания видеопотока с записывающего устройства в режиме реального времени, его последующее

разбиение на кадры и проведение аналитической обработки считываемого видеофайла, разбиваемого на кадры. За считывание видеопотока отвечает библиотека OpenCV. Листинг считывания данных, разбиения видеопотока на кадры и вызова функции аналитической обработки приведен на рис. 12.

```
def start_web(self):
        is_yawning = False
        yawning_true = False
        y_times = []
        time_s = time.time()
        video = cv2.VideoCapture(0)
        while True:
            frame = video.read()[1]
            gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
            faces = detector(gray)
            for face in faces:
              x, y = face.left(),face.top()
              width, height = face.right() - x, face.bottom() - y
                landmarks = predictor(gray, face)
                if yawning_true:
                    cv2.putText(frame, 'Asleep', (650, 100),
cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 3)
                elif yawning_detect(list_of_points[3], landmarks):
                    if not is_yawning:
                        y_times.append(time.time())
                        is_yawning = True
                        if (len(y_times) >= 3) and (y_times[-1] - y_times[-
3] <= 60):
                            yawning_true = True
                else:
                    is_yawning = False
                if blinking_detect(list_of_points[0], list_of_points[1],
landmarks) or \
                        glance_detect(list_of_points[0], list_of_points[1],
landmarks, gray) or \
                        head_detect(list_of_points[2], landmarks) or \
                     blinking_recognition(frame, x, y, width, height):
                    if time.time() - time_s >= 3:
                        cv2.putText(frame, 'ALARM!', (50, 100),
cv2.FONT_HERSHEY_PLAIN, 2, (0, 0, 255), 3)
                else:
                    time_s = time.time()
            cv2.imshow("Detector", frame)
            cv2.waitKey(1)
```

Рис.12. Листинг считывания данных, разбиения видеопотока на кадры и вызова функции аналитической обработки

Обе рассмотренные функции выполняют одну и ту же задачу для разных вариантов получения входных данных программой. Последним действием обеих функций является аналитическая обработка кадров, по-

лученных в результате разбиения видеофайла и отображение результатов пользователю.

После разбиения каждый кадр анализируется детектором лиц, который является встроенной функцией библиотеки dlib. После этого, на найденном детектором лице определяются 68 ключевых точек с помощью предварительно обученной модели нейронной сети shape_predictor_68_face_landmarks.dat.

Следующим действием является определение зевания. Для определения зевания человека на изображении используется функция yawning_detect(). Данная функция принимает на вход два аргумента - массив индексов точек, необходимых для определения зевания и полный набор ключевых точек, определенных моделью нейронной сети shape_predictor_68_face_landmarks.dat. Как было описано в теоретической части, для определения зевания нам необходимо знать двенадцать точек, шесть из которых отвечают за верхнюю губу, оставшиеся шесть - за нижнюю. Функция yawning_detect() определяет координаты нужных точек (значения по осям абсцисс и ординат) и поочередно вычисляет координаты середин верхней и нижней губы. Координаты данных точек вычисляются как средние значения координат ключевых точек обеих губ по обеим осям с использованием функции mean() библиотеки numpy. Численная разница между координатами по оси абсцисс средних точек верхней и нижней губ является первым катетом, по оси ординат – вторым. В таком случае, расстояние между средними точками губ является гипотенузой и высчитывается по теореме Пифагора. Использовать в качестве расстояния только численную разницу между координатами по оси ординат не корректно, так как данный способ будет выдавать неверный результат при наклоне головы. Затем полученный результат расстояния между губами сравнивается с пороговым значением для зевания, вычисленным автоматически при тестировании работы определения расстояния между губами на примерах изображений с зевающими и не зевающими людьми. Функция возвращает

значение True в случае, если на изображении было выявлено зевание и False, если зевание выявлено не было. Листинг функции определения зевания приведен на рис. 13.

```
def yawning_detect(lip_points, landmarks):
    top_lip_points = [[], []]
    for point in lip_points[0]:
        top_lip_points[0].append(landmarks.part(point).x)
        top_lip_points[1].append(landmarks.part(point).y)

    bottom_lip_points = [[], []]
    for point in lip_points[1]:
        bottom_lip_points[0].append(landmarks.part(point).x)
        bottom_lip_points[1].append(landmarks.part(point).y)

    top_lip_middle = (numpy.mean(top_lip_points[0]),
numpy.mean(top_lip_points[1]))
    bottom_lip_middle = (numpy.mean(bottom_lip_points[0]),
numpy.mean(bottom_lip_points[1]))

    distance = hypot(top_lip_middle[0] - bottom_lip_middle[0],
top_lip_middle[1] - bottom_lip_middle[1])

    return distance > 40
```

Рис. 13. Листинг функции определения зевания

В случае, если функция определения зевания вернула истинное значение три и более раз во временном промежутке в одну минуту, пользователю выводится на экран сообщение с предупреждением о том, что у человека на видео выявлено состояние сонливости и управление транспортным средством небезопасно.

Далее следует определение признаков потери концентрации у водителя. Для этого поочередно вызываются функции определения закрытости глаз, отвлеченного взгляда и поворота головы.

Закрытость глаз у человека на изображении определяется двумя способами. Первый способ заключается в использовании самостоятельно реализованной и обученной нейронной сети. Для этого вызывается функция blinking_recognition(). Данная функция принимает в качестве аргументов изображение, координаты верхней левой вершины прямоугольника, со-

держащего определенное детектором лицо, и длины сторон данного прямоугольника. В первую очередь функция предварительно обрабатывает полученное изображение, выделяя для работы нейронной сети только область глаз, составляющая 40 % от высоты прямоугольника с верхним отступом в 10 % и нижним в 50 %. Полученная в результате область глаз переводится в черно-белую цветовую гамму, а ее размер изменяется в соответствии с требованиями для корректной работы модели нейронной сети, то есть 100x40 пикселей. Далее модель нейронной сети анализирует полученное в результате предварительной обработки изображение области глаз и выдает результат, значением которого является класс, к которому нейронная сеть отнесла данное изображение (0 – глаза закрыты, 1 – глаза открыты). В соответствии с результатом работы нейронной сети создается переменная, которой присваивается значение True в случае, если были распознаны закрытые глаза и False в ином случае. Данная переменная возвращается в качестве результата выполнения функции blinking_recognition(). Листинг данной функции представлен на рис. 14.

```
def blinking_recognition(face, x, y, w, h):
    eye_area = face[int((y + h) / 10):int((y + h) / 2), x:x + w]
    eye_area = cv2.resize(eye_area, (100, 40))
    if len(eye_area.shape) == 3 and eye_area.shape[-1] == 3:
        eye_area = cv2.cvtColor(eye_area, cv2.COLOR_RGB2GRAY)
        eye_area = numpy.expand_dims(eye_area, -1)
    elif len(eye_area.shape) == 3 and eye_area.shape[-1] == 4:
        eye_area = cv2.cvtColor(eye_area, cv2.COLOR_RGBA2GRAY)
        eye_area = numpy.expand_dims(eye_area, -1)
    if len(eye_area.shape) == 2:
        eye_area = numpy.expand_dims(eye_area, -1)

    eye_area_arr = [eye_area]
    eye_area_arr = numpy.array(eye_area_arr)
    predictions = model.predict(eye_area_arr)
    verdict = predictions.argmax()

    result = False
    if verdict == 0:
        result = True
    return result
```

Рис. 14. Листинг функции определения закрытости глаз

Второй способ определения закрытости глаз у человека на изображении заключается в использовании ключевых точек, определенных моделью нейронной сети shape_predictor_68_face_landmarks.dat. Для этого вызывается функция blinking_detect(). Данная функция принимает на вход три аргумента – два массива индексов точек, необходимых для определения закрытости глаз и полный набор ключевых точек. Критерием закрытости глаз является сомкнутость век, что, в свою очередь характеризуется малым расстоянием между средними точками нижнего и верхнего век обоих глаз. Однако, экспериментально было установлено, что определения расстояния между серединами век недостаточно для корректного определения закрытости глаз в следствие большого разнообразия размеров и форм глаз у разных людей. Для наиболее точного определения закрытости глаз нам необходимо высчитывать отношение высоты глаза (расстояния между серединами век) к его ширине (длине оси глазной щели). Определение высоты и ширины глаза аналогично определению расстояния между верхней и нижней губами из алгоритма, рассмотренного ранее.

В первую очередь в функции blinking_detect() определяется отношение ширины и высоты для левого и правого глаза, после чего определяется среднее значение арифметическое этих отношений. Затем полученный результат сравнивается с пороговым значением для закрытости глаз, вычисленным автоматически при тестировании работы определения отношения высоты и ширины глаза на примерах изображений с людьми с закрытыми и открытыми глазами. Функция возвращает значение True в случае, если на изображении была определена закрытость глаз и False, если закрытость глаз выявлена не была. Листинг алгоритма определения закрытости глаз приведен на рис. 15.

Для определения отвлеченного взгляда у человека на изображении используется функция glance_detect(). Данная функция принимает на вход четыре аргумента - два массива индексов точек для левого правого глаза, полный набор ключевых точек и изображение в черно-белой цветовой

гамме. Функция glance_detect() по ключевым точкам для каждого из глаз определяет его область на изображении, выделяет эту область и переводит выделенную область в бинарную черно-белую гамму, где каждый пиксель либо черного, либо белого цвета, без оттенков серого. Далее область глаза делится посередине на две части - правую и левую, после чего подсчитывается количество белых пикселей на каждой из сторон.

```
def get_ratio(points, landmarks):
    left_point = (landmarks.part(points[0]).x, landmarks.part(points[0]).y)
    right_point = (landmarks.part(points[3]).x, land-
marks.part(points[3]).y)
    center_top_point = ((int(landmarks.part(points[1]).x + land-
marks.part(points[2]).x) / 2),
                        int((landmarks.part(points[1]).y + land-
marks.part(points[2]).y) / 2))

    center_bottom_point = ((int(landmarks.part(points[4]).x + land-
marks.part(points[5]).x) / 2),
                        int((landmarks.part(points[4]).y + land-
marks.part(points[5]).y) / 2))

    widht = hypot((left_point[0] - right_point[0]), (left_point[1] -
right_point[1]))
    height = hypot((center_top_point[0] - center_bottom_point[0]), (cen-
ter_top_point[1] - center_bottom_point[1]))

    return widht / height


def blinking_detect(points_left, points_right, landmarks):
    ratio_left_eye = get_ratio(points_left, landmarks)
    ratio_right_eye = get_ratio(points_right, landmarks)
    is_blinking = (ratio_left_eye + ratio_right_eye) / 2 > 4.8
    return is_blinking
```

Рис. 15. Листинг алгоритма определения закрытости глаз

В случае, если зрачок расположен не по середине видимой части глазного яблока, количество белых пикселей с одной из сторон будет значительно больше, чем на другой. Для наиболее точного определения положения зрачка подсчитывается отношение количества белых пикселей на левой стороне глаза и количества белых пикселей на правой стороне глаза. Полученный результат сравнивается с пороговыми значениями для отвлеченности взгляда, вычисленными автоматически при тестировании работы

определения отвлеченности взгляда. В случае, если определена отвлеченность взгляда, функция glance_detect() возвращает значение True. В противном случае функция glance_detect() возвращает значение False. Листинг алгоритма определения отвлеченности взгляда приведен на рис. 16.

```
def get_ratio(points, landmarks, grey):
    eye_area = np.array([(landmarks.part(points[0]).x, land-
marks.part(points[0]).y), (landmarks.part(points[1]).x, land-
marks.part(points[1]).y), (landmarks.part(points[2]).x, land-
marks.part(points[2]).y), (landmarks.part(points[3]).x, land-
marks.part(points[3]).y), (landmarks.part(points[4]).x, land-
marks.part(points[4]).y), (landmarks.part(points[5]).x, land-
marks.part(points[5]).y)], np.int32)
    height = grey.shape[0]
    width = grey.shape[1]
    mask = np.zeros((height, width), np.uint8)
    cv2.polylines(mask, [eye_area], True, 255, 2)
    cv2.fillPoly(mask, [eye_area], 255)
    eye = cv2.bitwise_and(grey, grey, mask=mask)
    start_y = np.min(eye_area[:, 1])
    end_y = np.max(eye_area[:, 1])
    start_x = np.min(eye_area[:, 0])
    end_x = np.max(eye_area[:, 0])
    grey_eye = eye[start_y: end_y, start_x: end_x]
    threshold_eye = cv2.threshold(grey_eye, 70, 255, cv2.THRESH_BINARY)[1]
    height, width = threshold_eye.shape
    left_half_threshold = threshold_eye[0: height, 0: int(width / 2)]
    right_half_threshold = threshold_eye[0: height, int(width / 2): width]
    left_half_white_part = cv2.countNonZero(left_half_threshold)
    right_half_white_part = cv2.countNonZero(right_half_threshold)
    try:
        return left_half_white_part / right_half_white_part
    except:
        return 0


def glance_detect(points_left, points_right, landmarks, grey):
    left_eye_glance_direction = get_ratio(points_left, landmarks, grey)
    right_eye_glance_direction = get_ratio(points_right, landmarks, grey)
    is_glance_distracted = (left_eye_glance_direction +
right_eye_glance_direction < 1) or \
            (left_eye_glance_direction + right_eye_glance_direction > 5)

    return is_glance_distracted
```

Рис. 16. Листинг алгоритма определения отвлеченности взгляда

Для определения поворота головы у человека на изображении используется функция head_detect(). Данная функция принимает в качестве входных данных два аргумента - массив индексов точек, необходимых для

определения положения головы и полный набор ключевых точек. Для определения поворота головы нам необходимы три ключевые точки, отвечающие за кончик носа, левый и правый виски. В первую очередь функция head_detect() определяет расстояния от кончика носа до каждого из висков. В случае, если голова повернута в какую-либо из сторон, расстояние от кончика носа до одного из висков значительно больше расстояния от кончика носа до другого виска. Для корректного определения поворота головы достаточно просчитывать расстояние от координаты по оси абсцисс кончика носа до координат по оси абсцисс каждого из висков. Отношение полученных значений расстояний является характеристикой поворота головы. Полученный результат сравнивается с пороговыми значениями для поворота головы, вычисленными автоматически при тестировании работы определения поворота головы. В случае, если определен поворот головы в какую-либо сторону, функция head_detect() возвращает значение True. В противном случае функция head_detect() возвращает значение False. Листинг алгоритма определения поворота головы приведен на рис. 17.

```
def head_detect(head_points, landmarks):
    left_point = landmarks.part(head_points[0]).x
    right_point = landmarks.part(head_points[2]).x
    center_point = landmarks.part(head_points[1]).x

    left_half_width = abs(left_point - center_point)
    right_half_width = abs(right_point - center_point)
    try:
        head_direction = left_half_width / right_half_width
    except:
        head_direction = 0
    is_head_turned = head_direction < 0.4 or head_direction > 2.5
    return is_head_turned
```

Рис. 17. Листинг алгоритма определения поворота головы

Если хотя бы одна из функций определения потери концентрации возвращает истинное значение на протяжении трех и более секунд, пользователю выводится на экран сообщение, предупреждающее его о том, что у человека на видео выявлены признаки потери концентрации, описанные выше, и управление транспортным средством является небезопасным.

4.4. Реализация интерфейса приложения.

Для комфортного взаимодействия пользователя с системой был разработан графический интерфейс. Для разработки интерфейса были использованы графическая библиотека PyQt5 и приложение QtDesigner. Внешний вид графического интерфейса пользователя приведен на рис. 18.

Рис. 18. Внешний вид графического интерфейса пользователя

Выводы по четвертой главе

В данной главе были описаны средства разработки системы, приведено описание процесса формирования обучающей выборки, реализации нейронной сети в соответствии с заранее определенной топологией, обучения и тестирования построенной нейронной сети. Также описан процесс программной реализации алгоритмов распознавания признаков утомления и потери концентрации у человека на изображении на основе работы предварительно обученной модели нейронной сети shape_predictor_68_face_landmarks.dat и создания графического интерфейса приложения.

5.ТЕСТИРОВАНИЕ

5.1. Тестирование нейронной сети

В ходе выполнения работы была составлена обучающая и контрольная выборки. Тестирование проводилось на изображениях, ранее не участвовавших в обучении.

Во время тестирования нейронная сеть смогла определить признаки утомления и потери концентрации у водителя на изображении с точностью в 89 %.

5.2. Функциональное тестирование

Функциональное тестирование – тестирование, направленное на проверку корректности работы функциональности приложения (корректность реализации функциональных требований) [15].

Было проведено тестирование на основе функциональных требований. Ниже представлен протокол функционального тестирования.

Тест № 1. Цель: Проверить возможность запуска анализа видеопотока, записываемого в режиме реального времени.

Действие: Пользователь нажимает на кнопку «Начать запись».

Ожидаемый результат: Открылось окно записи, в котором пользователь наблюдает записываемый с веб-камеры видеопоток, дополненный выводимыми на него результатами работы распознавателя признаков потери концентрации.

Тест пройден? Да

Тест № 2. Цель: Проверить возможность выбора видеофайла.

Действие: Пользователь нажмите на кнопку «Выбрать файл».

Ожидаемый результат: Открылся проводник с возможностью выбора файла для анализа.

Тест пройден? Да

Тест № 3. Цель: Проверить возможность анализа выбранного заранее записанного видеофайла.

Действие: В открытом проводнике пользователь выбирает файл и нажимает на кнопку «open».

Ожидаемый результат: Открылось окно воспроизведения, в котором пользователь наблюдает воспроизведение выбранного заранее записанного видеофайла, дополненное выводимыми на него результатами работы распознавателя признаков потери концентрации.

Тест пройден? Да

Выводы по пятой главе

В пятой главе было произведено тестирование работы всей системы. Было проведено тестирование работы нейронной сети. Также были подготовлены тесты для функционального тестирования, после чего было проведено функциональное тестирование реализованного системы распознавания признаков утомления и потери концентрации.

ЗАКЛЮЧЕНИЕ

В рамках данной работы была разработана система распознавания признаков усталости и потери концентрации у водителя во время управления транспортным средством в режиме реального времени на основании видео. При этом были решены следующие задачи.

1. Проведен анализ предметной области и аналогичных решений.
2. Разработаны алгоритмы для распознавания потери концентрации на изображении.
3. Спроектирована топология нейронной сети.
4. Осуществлен сбор и предобработка данных для обучения.
5. Реализована и обучена нейронную сеть.
6. Реализовано и протестировано приложение для распознавания признаков утомления и потери концентрации у человека на видео.

ЛИТЕРАТУРА

1. Begino Y., Goodfellow I., Courville A. Deep Learning. – MIT Press, 2016. – 773 p.

2. Dalal N., Triggs B. Histograms of Oriented Gradients for Human. // Detection IEEE, 2005. – P. 886–893.

3. Driver Alert Control. [Электронный ресурс] URL: https://www.media.volvocars.com/ru/ru-ru/media/videos/12261/ (дата обращения: 13.01.2019).

4. Keras. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/Keras/ (дата обращения: 21.02.2019).

5. Mercedes-Benz Attention Assist. [Электронный ресурс] URL: https://www.mbusa.com/mercedes/benz/safety/ (дата обращения: 13.01.2019).

6. OpenCV. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/OpenCV/ (дата обращения: 21.02.2019).

7. Seeing Machines by General Motors. [Электронный ресурс] URL: https://www.seeingmachines.com/blog/2018/04/17/seeing-machines-fovio-driver-monitoring-technology-features-in-gm-cadillac-ct6-super-cruise/ (дата обращения: 13.01.2019).

8. Shu C., Ding X., Fang C. Histogram of the oriented gradient for face recognition. // Tsinghua Science and Technology, 2011. – No. 2 (16). – P. 216–224.

9. TensorFlow. [Электронный ресурс] URL: https://ru.wikipedia.org/wiki/TensorFlow/ (дата обращения: 21.02.2019).

10. Документация Keras. [Электронный ресурс] URL: http://keras.io/ (дата обращения: 21.02.2019).

11. Документация OpenCV. [Электронный ресурс] URL: http://opencv.org/ (дата обращения: 21.02.2019).

12. Документация TensorFlow. [Электронный ресурс] URL: https://www.tensorflow.org/ (дата обращения: 21.02.2019).

13. Кафтанников И.Л., Парасич А.В. Проблемы формирования обучающей выборки в задачах машинного обучения. // Вестник ЮУрГУ, 2016. – 10 с.

14. Классификация. [Электронный ресурс] URL: http://www.machinelearning.ru/wiki/index/php&title=Классификация/ (дата обращения: 12.03.2019).

15. Куликов С.С. Тестирование программного обеспечения. Базовый курс. – Издательство "Четыре Четверти", 2015. – 294 с.

16. Овчаренко М. С. Анализ и прогноз состояния и уровня аварийности на дорогах Российской Федерации и пути по ее снижению. // Научно-методический электронный журнал «Концепт». – 2016. – Т. 15. – С. 1661–1665.

17. Официальный сайт dlib. [Электронный ресурс] URL: https://dlib.net/ (дата обращения: 08.03.2019).

18. Под Рязанью заснувший водитель спровоцировал ДТП со смертельным исходом. [Электронный ресурс] URL: https://newizv.ru/news/incident/pod-ryazanyu-zasnuvshiy-voditel-sprovotsiroval-dtp-so-smertelnym-ishodom/ (дата обращения: 12.09.2018).

19. Романов В.П. Интеллектуальные информационные системы в экономике. Учебное пособие. – Издательство "ЭКЗАМЕН", 2003. – 494 с.

20. Фаулер М. Основы UML. Краткое руководство по стандартному языку объектного моделирования, 2005. – 185 с.