МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

02.03.03 Математическое обеспечение и администрирование информационных систем
(код и наименование направления подготовки, специальности)

Технология программирования
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка алгоритма для распознавания эмоций на основе анализа изображений»

Студент А.В. Рытов
(И.О. Фамилия) (личная подпись)

Руководитель к.т.н., доцент, В.С. Климов
(ученая степень, звание, И.О. Фамилия)

Консультант М.В. Дайнеко

Тольятти 2020

АННОТАЦИЯ

Тема бакалаврской работы: «Разработка алгоритма для распознавания эмоций на основе анализа изображений».

В данной бакалаврской работе исследуется вопрос распознавания эмоций при помощи технологий компьютерного зрения, а также разрабатывается соответствующий алгоритм, после чего оценивается его точность.

Объект исследования – алгоритм распознавания эмоций на изображении.

Предмет исследования – анализ точности алгоритма распознавания эмоций.

Цель исследования – разработка алгоритма для распознавания эмоций на основе анализа изображений.

Структура бакалаврской работы представлена введением, тремя разделами, заключением и списком литературы.

Во введении описывается актуальность проводимого исследования, формулируется цель и ставятся задачи, которые необходимо решить.

В первом разделе рассматривается сущность задачи распознавания эмоций, а также приводятся наиболее известные существующие аналоги.

Во втором разделе предлагаются и рассматриваются решения необходимые при проектировании алгоритма, формируется его конечный вариант, а также проектируется математическая модель алгоритма.

В третьем разделе рассматриваются язык программирования, используемый для реализации алгоритма, необходимые библиотеки и технологии, размещаются наиболее важные фрагменты кода, а также проводится тестирование точности данного алгоритма.

В заключении представляются выводы по проделанной работе.

В работе использовано 25 формул, 20 рисунков, список литературы содержит 20 литературных источников. Общий объем выпускной квалификационной работы составляет 54 страницы.

ABSTRACT

The present graduation work deals with developing an algorithm for recognizing emotions on the basis of image analysis.

Much attention is given to the issue of emotion recognition by means of the computer vision technologies. We also give full coverage to developing the appropriate algorithm, after which its accuracy is assessed.

The object of the research is the algorithm for recognizing emotions in an image.

The subject of the research is the analysis of the algorithm accuracy for recognizing emotions.

The purpose of the study is to develop an algorithm for recognizing emotions on the basis of the images analysis.

The graduation work consists of an introduction, three chapters, conclusions and a list of references.

In the introduction, the relevance of the conducted research is explained, the purpose and the tasks to be solved are stated.

The first chapter considers the essence of the problem of recognizing emotions, and presents the most widespread and common existing analogues.

In the second section, the solutions necessary for algorithm design are suggested and discussed, its final variant is formed, and a mathematical model of the algorithm is designed.

The third chapter dwells on the programming language used to implement the algorithm, the necessary libraries and technologies, the most important code fragments are revealed, as well as the accuracy of this algorithm is evaluated and checked.

In conclusion, it is pointed out what tasks have been accomplished during the work on the algorithm, and the ways of its further development are described.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В настоящее время, когда человечество переступило порог двадцатого века и вступило в третье тысячелетие нашей эры, информационные технологии стали неотъемлемой частью человеческой жизни. Их грамотное применение на производственном поприще является целью любого профессионала. ЮНЕСКО определяет информационные технологии как комплекс взаимосвязанных технологических, научных, инженерных дисциплин, которые направлены на хранение, обработку и использование информации. ИТ включают в себя множество разнообразных областей развития. Например, интеллектуальные системы, когнитивные ИТ, технологии баз данных, компьютерная графика, управленческие информационные системы, а также многие другие. Одной из таких областей является компьютерное зрение.

Компьютерное зрение – достаточно молодая и быстро развивающаяся область информационных технологий. Как самостоятельная дисциплина компьютерное зрение появилось в начале 1950-х годов, когда John von Neumann предложил анализировать микроснимки при помощи компьютеров путём сравнения яркости соседних частей изображения. Основной целью данной дисциплины является построение систем, способных извлекать из изображений полезную информацию об объектах окружающей среды и её дальнейшее использование. Обработка информации может осуществляться как на универсальных, так и на специализированных компьютерах. С помощью компьютерного зрения можно решать множество различных задач. Например, сегментация изображений, обнаружение объектов, классификация изображений, распознавание динамических объектов и многие другие. Одной из таких задач является распознавание эмоций на изображении. Именно данная задача и будет реализовываться в ходе выполнения данной выпускной квалификационной работы.

Актуальность данной работы заключается в том, что тема распознавания эмоций быстро развивается, при помощи эмоций можно

определить какие действия в той или иной ситуации буду приемлемы по отношению к человеку, а какие нет. Поэтому обучив компьютер распознаванию эмоций можно значительно оптимизировать и автоматизировать некоторые процессы.

Объектом исследования является алгоритм распознавания эмоций.

Предметом исследования является анализ точности алгоритма распознавания эмоций.

Целью работы является разработка алгоритма для распознавания эмоций на изображении, при этом средняя точность классификации эмоций должна быть больше 77%, а распознавание лиц должно производиться при их отклонении от фронтального положения вплоть до 45 градусов.

В соответствии с обозначенной целью были сформулированы следующие задачи:

– проанализировать предметную область;

– подготовить и рассмотреть альтернативы и решения, которые необходимо использовать при проектировании алгоритма;

– выбрать язык программирования и технологию необходимые для реализации принятых решений;

– исходя из принятых решений реализовать алгоритм для распознавания эмоций на изображении на выбранном языке с использованием выбранной технологии;

– добавить возможность распознавания эмоций на видеопотоке, получаемом с веб-камеры;

– провести анализ эффективности разработанного алгоритма.

При подготовке бакалаврской работы была опубликована статья по теме разработка алгоритма для распознавания эмоций на изображении. Результаты работы были доложены на VI Международной научно-практической конференции молодых ученых «Прикладная математика и информатика: современные исследования в области естественных и технических наук».

1 АНАЛИЗ СОСТОЯНИЯ ВОПРОСА РАСПОЗНАВАНИЯ ЭМОЦИЙ НА ИЗОБРАЖЕНИИ

1.1 Введение в задачу распознавания эмоций на изображении

На протяжении жизни каждый человек испытывает какие-либо эмоции, исходя из которых другие люди могут догадаться что он сейчас чувствует и какие действия будут приемлемы по отношению к нему. Распознавание эмоций одно из наиболее развивающихся направлений в сфере искусственного интеллекта. Данную технологию можно применить в различных сферах человеческой жизни. Например, распознавать состояние водителя, находящегося за рулём автомобиля, наблюдать за обучающимися, которые проходят дополнительные курсы, или программный модуль встроенный в камеры видеонаблюдения технологии «умный дом» может оценивать эмоциональное состояние хозяина и в зависимости от этого предпринимать дальнейшие действия по его повышению. Из выше изложенного можно сделать вывод, что эмоции – это очень важная и неотъемлемая часть жизни каждого человека. Эмоция – это психический процесс средней продолжительности, отражающий субъективное оценочное отношение к существующим или возможным ситуациям и объективному миру. Эмоции могут быть выражены самыми разнообразными способами: мимикой, позой, двигательными реакциями и другими способами. Однако наиболее распространённым и наиболее понятным способом передачи эмоций является лицо человека. Известный американский психолог Paul Ekman в ходе своих исследований в середине 1970-х годов установил, что 7 человеческих эмоций являются универсальными и они могут быть поняты человеком независимо от его народности и культуры. Данные эмоции называются базовыми.

К базовым эмоциям относятся следующие:

- гнев,
- страх,

– печаль,
– отвращение,
– презрение,
– удивление,
– радость.

Для того чтобы показать, как выглядят данные эмоции на лице человека ниже будет приведён рисунок 1.

Рисунок 1 – Базовые эмоции человека

На рисунке, расположенном выше, можно видеть, как меняется лицо человека в зависимости от того какую из 7 базовых эмоций он испытывает.

Стоит отметить, что эмоцию типа презрение современные психологи считают не чувством, а отношением, то есть в большей степени она выражается не с помощью мимики лица, а с помощью действий, направленных на объект презрения. Как показано на рисунке 1, когда человек испытывает эмоцию презрение его мимика изменяется совсем незначительно.

В связи с этим в ходе разработки алгоритма распознавания эмоций будут учитываться все базовые эмоции за исключением данной. Вместо этого будет распознаваться состояние отсутствия отчётливо выраженных эмоций. Данное состояние будет называться – спокойствие.

Для распознавания эмоций на изображении существует 2 подхода: классический и с помощью «глубокого обучения».

Классический подход заключается в том, что на изображении, распознаётся лицо человека для этого можно использовать метод Виолы-Джонса или классификатор HOG после чего в выделенной области по методу ключевых точек размечают от 5 до 68 точек, для того чтобы с их помощью можно было захватить мимику лица человека [1]. Они привязываются к положению бровей, глаз, губ, носа, челюсти. Данный метод можно реализовать, например, с помощью свёрточной нейронной сети (CNN). Далее происходит нормализация полученных значений и после чего они подаются на вход классификатору, например, методу опорных векторов (SVM) или классификатору случайный лес (Random Forest). Также для повышения точности системы можно в ключевых точках вычислить дескрипторы, например, SIFT или LBP, и объединить их, тем самым будет получен вектор признаков с повышенной точностью по сравнению с исходным. Данный подход является устаревшим и менее точным, поэтому при разработке алгоритма будет использоваться «глубокое обучение» [5].

Распознавание эмоций с помощью «глубокого обучения», является более точным, но более ресурсозатратным подходом. В распознаватель лиц, который может быть представлен алгоритмом Виолы-Джонса, обратной свёрточной нейронной сетью (DNN) или детектором объектов с максимальным запасом (MMOD) подаётся исходное изображение, он вычисляет область в которой находится лицо человека, далее исходное изображение переводиться из цветового формата RGB или BGR в оттенки серого, для того чтобы уменьшить признаковое пространство и повысить скорость работы классификатора эмоций. После чего область, найденная с

помощью распознавателя лиц в оттенках серого, нормализуется и подаётся на вход классификатору эмоций, которым является обученная свёрточная нейронная сеть (CNN). В свою очередь CNN возвращает критерии силы 6 базовых эмоций и состояния спокойствия [7]-[8].

Исходя из выше сказанного наиболее важными задачами при проектировании алгоритма для распознавания эмоций являются:

– выбор распознавателя лиц;

– выбор классификатора эмоций.

Однако прежде чем приступить к проектированию нового алгоритма необходимо познакомиться с уже имеющимися аналогами.

1.2 Обзор существующих аналогов

Рассмотрение некоторых существующих аналогов следует начать с приложения FaceReader от голландской компании Noldus Information Technology, так как оно на сегодняшний день является самым точным и с минимальным временем обработки кадров.

FaceReader – это самая надежная автоматизированная система для распознавания эмоций на изображениях, включающая шесть основных или базовых эмоций: счастье, печаль, гнев, удивление, страх и отвращение. Кроме того, FaceReader может распознавать "нейтральное" состояние и анализировать "презрение".

Независимо от того, является ли участник теста ребенком, взрослым или пожилым человеком, FaceReader настраивает анализ на модель, которая лучше всего подходит для данного исследования.

Работу FaceReader можно описать следующими шагами:

– вычисление области на изображении, в которой расположено лицо, с помощью алгоритма Виолы-Джонса;

– проектирование точечной 3D-модели лица с использование более 500 ключевых точек сетки;

– обработка модели с помощью «глубокого обучения»;

– классификация эмоций с помощью искусственной нейронной сети (ANN).

Далее на рисунке 2 будет представлен пример работы приложения FaceReader.

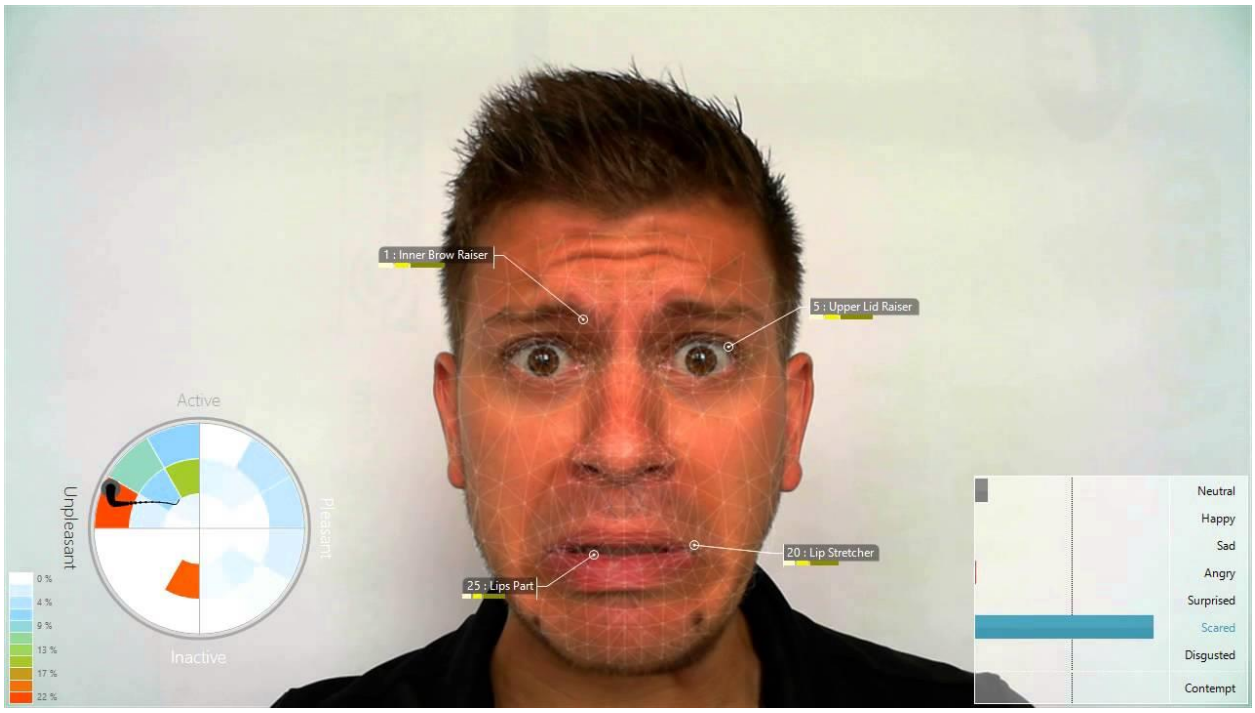

Рисунок 2 – Пример работы приложения FaceReader

На рисунке 2 можно видеть, как приложение FaceReader справляется со своей задачей. Кроме простого определения эмоций оно также выделяет области лица исходя из которых была выявлена преобладающая эмоция, накладывает 3D-сетку на лицо и выполняет некоторые другие действия.

Преимуществами данного приложения являются:

– средний процент распознавания эмоций равен 93, и он не зависит от положения лица в пространстве;

– поддержка большого количества кодеков видеофайлов таких как MPEG1, MPEG2, DV-AVI и множество других, к тому же программа может обрабатывать видеопоток, поступающий с веб-камеры пользователя, а также данное приложение поддерживает обработку отдельных изображений;

– поддержка различных возможностей визуализации таких как построение диаграмм, гистограмм, наложение сетки и отмечание областей особого внимания на лице.

Недостатком данного приложения является то, что в случае наличия на лице человека очков распознавание может происходить неточно.

Следующим приложением, которое также является аналогом разрабатываемого алгоритма, является Xeoma, а именно модуль Детектор лиц (Эмоции).

Xeoma – это программа для видеонаблюдения с простым интерфейсом, гибкими настройками, качественным сервисом и профессиональными функциями, с помощью неё можно получить доступ к видеокамерам абсолютно из любой точки мира. Разработчиком данной программы является российская компания ФеленаСофт. Данное приложение является лидером продаж среди программ, используемых для видеонаблюдения.

Начиная с версии 18.11.21 в данную программу входит программный модуль Детектор лиц (Эмоции). Он может анализировать изображения, находить на них лица и по ним распознавать 6 базовых эмоций: счастье, удивление, раздражение (гнев), отвращение, испуг (страх) и печаль. Также данное приложение может распознавать состояние – нейтральность (спокойствие).

Преимуществами данного приложения являются:

– удобный и достаточно простой графический интерфейс;

– гибкие настройки классификатора эмоций;

– возможность распознавать эмоций с помощью пользовательской веб-камеры и на видеопотоке из видеофайла, так как данная программа также, как и FaceReader поддерживает множество кодеков, а также возможность загружать обычные изображения или видеопоток с удалённой веб-камеры;

– возможность распознавать эмоции сразу нескольких людей на изображении.

Недостатком данного приложения является, то что присутствует не точное распознавания эмоций, иногда распознаются эмоций, которые человек не испытывает.

На рисунке 3 представленном ниже будет продемонстрирован пример работы приложения Xeoma для распознавания лиц.

Рисунок 3 – Пример распознавания эмоций на лице в Xeoma

На данном рисунке можно видеть, что модуль Детектор лиц (Эмоции) приложения Xeoma, не смотря на присутствие на изображении объектов, мешающих распознаванию не только эмоций, но и лица (данными объектами являются борода и капюшон), справился со своей задачей и правильно распознал преобладающую эмоцию, но всё же можно заметить, что данный

модуль обнаружил на лице этого человека такие эмоции как счастье и испуг, чего конечно же на данном изображении нет.

На рисунке 4 расположенном ниже будет представлен результат распознавания эмоций на изображении, где представлены лица двух людей.

Рисунок 4 – Пример распознавания эмоций на лицах в Xeoma

На рисунке, расположенном выше, можно видеть, что модуль Детектор лиц (Эмоции) приложения Xeoma смог распознать эмоции сразу нескольких человек, что несомненно является его преимуществом перед другими аналогами, которые могут распознавать эмоции только одного человека.

Последним аналогом, который хотелось бы рассмотреть в ходе выполнения данной выпускной квалификационной работы, является EmoDetect.

EmoDetect – программа, разработанная российской компанией Нейроботикс, для регистрации (записи) лица человека и определения по нему эмоционального состояния. Классификатор программы EmoDetect

распознаёт до 20 информативных локальных признаков лица, характеризующих психоэмоциональное состояние человека (ASM). Данное приложение, как и его аналоги, может распознавать 6 базовых эмоций: радость (счастье), удивление, гнев, отвращение, страх и грусть (печаль), а также состояние нейтральность (спокойствие).

Также стоит отметить, что компания Нейроботикс начала свою деятельность в 2004 году в сфере разработки и производства оборудования для исследований в нейрофизиологии и психофизиологии. С тех пор Нейроботикс активно развивается и взаимодействует с различными международными компаниями. Выше сказанное говорит о том, что у Нейроботикс имеется достаточное количество опыта в области компьютерного зрения для разработки алгоритма распознавания человеческих эмоций на изображении с хорошей точностью.

Преимуществами данного приложения являются:

– возможность классификации эмоций 3 разнообразными классификаторами такими как нейронная сеть, система решающих правил и по взвешенной сумме признаков;

– возможность алгоритма адоптироваться под каждого человека индивидуально;

– возможность построения гистограмм и диаграмм для того чтобы более наглядно продемонстрировать как изменялось эмоциональное состояние человека с течением времени видеопотока;

– возможность распознавания эмоций при условии, что человек находиться в очках;

– предоставление программного интерфейса приложения для того чтобы подключить модули распознавания EmoDetect к другой программе.

Недостатком данного приложения является отсутствие возможности распознавания лица, повёрнутого в профиль.

На рисунке 5 который представлен ниже будет продемонстрирован пример работы приложения по распознаванию эмоций EmoDetect.

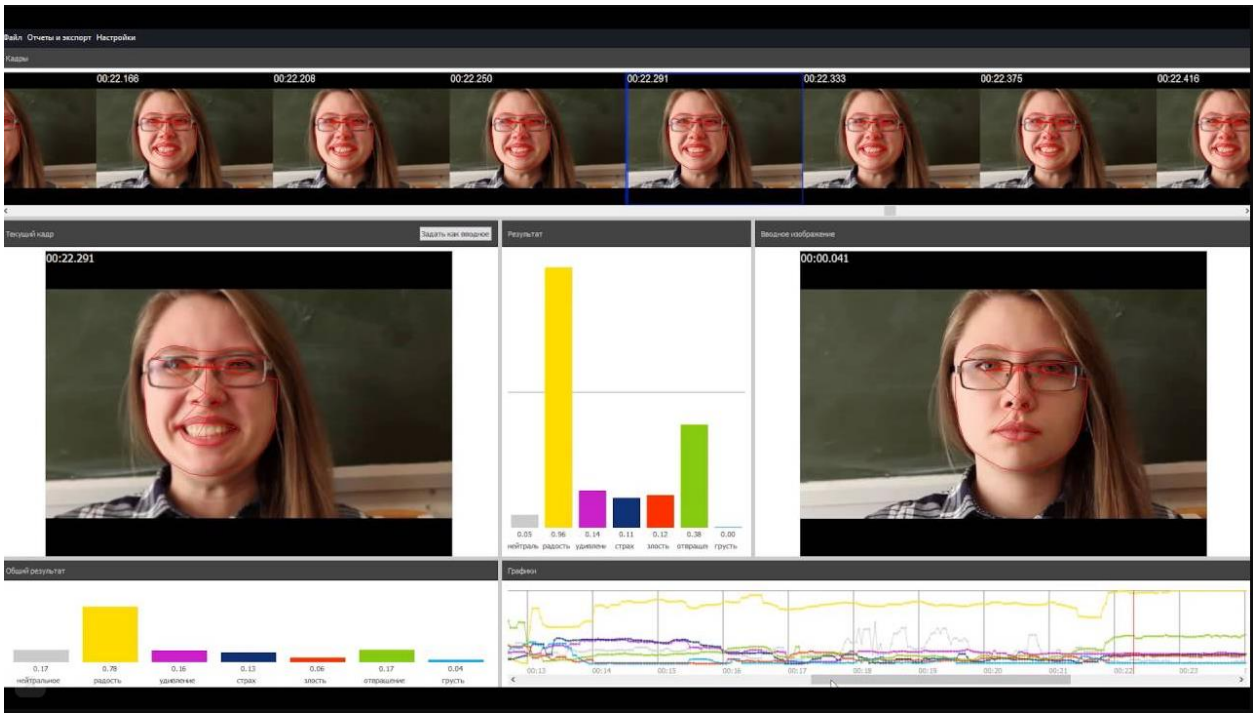

Рисунок 5 – Пример работы приложения EmoDetect

На рисунке 5 можно видеть, как приложение EmoDetect распознаёт эмоции на видеопотоке. Стоит отметить, что диаграммы эмоционального состояния человека создаются и изменяются в режиме реального времени. Также программа, как и FaceReader, размечает на лице человека ключевые точки исходя из которых распознаётся преобладающая эмоция.

В данном разделе была описана сущность задачи распознавания эмоций, рассмотрены подходы к её реализации, а также представлены некоторые наиболее известные аналоги разрабатываемого алгоритма.

2 ПРОЕКТИРОВАНИЕ АЛГОРИТМА ДЛЯ РАСПОЗНАВАНИЯ ЭМОЦИЙ НА ОСНОВЕ АНАЛИЗА ИЗОБРАЖЕНИЙ

2.1 Описание алгоритма

Ранее были обозначены 2 наиболее важных задачи, которые необходимо выполнить при проектировании алгоритма для распознавания эмоций. Далее будет рассмотрена первая из них – выбор распознавателя лиц.

Для того чтобы выделять лица людей на изображении было разработано большое число распознавателей, у каждого из которых есть свои особенности. В ходе данной выпускной квалификационной работы были выбраны наиболее известные из них:

- каскад классификаторов Хаара (алгоритм Виолы-Джонса);
- обратная свёрточная нейронная сеть (DNN);
- гистограмма направленных градиентов (HOG);
- детектор объектов с максимальным запасом (MMOD).

Каждый из них отличается скоростью работы, точностью распознавания и реакцией на окклюзию.

Окклюзия – ситуация, в которой два объекта расположены приблизительно на одной линии и один более близкий к веб-камере объект частично закрывает видимость другого объекта. В случае распознавания лиц окклюзия – это ситуация, когда лицо может быть частично закрыто каким-нибудь объектом, например, ладонью, шляпой, очками, бородой или множеством других объектов.

Далее будут рассмотрены каждый из выше упомянутых алгоритмов после чего будет произведено их сравнение.

Наиболее известным алгоритмом в области распознавания объектов несомненно является алгоритм Виолы-Джонса, основанный на использовании каскада классификатора Хаара. Он был разработан и представлен в 2001 году, его создателями являются Paul Viola и Michael Jones. В настоящее время существует множество разнообразных

модификаций данного алгоритма, но в данной работе будет рассмотрена наиболее эффективная из них представленная в библиотеке OpenCV.

Данный алгоритм основан на так называемом принципе сканирующего окна, то есть на изображении выделяется область, в которую попадает лишь часть исходного изображения, внутри она разбивается на так называемые ячейки, и после анализа одного положения области происходит сдвиг на одну ячейку, то есть изменение области, которая также анализируется. Таким образом сканируется всё изображение, размер сканирующего окна в ходе работы алгоритма может меняться. Суть анализа области тесно связана с признаками Хаара и интегральным представлением изображения.

На рисунке 6 представленном ниже, будут изображены некоторые признаки Хаара.

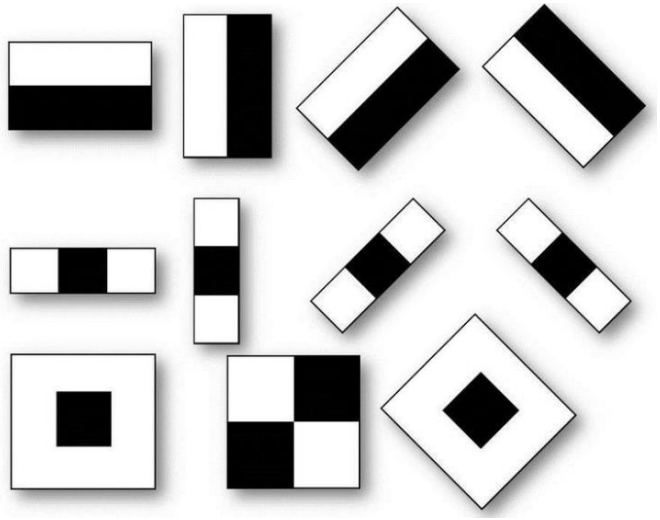

Рисунок 6 – Признаки Хаара

На данном рисунке можно видеть некоторые из признаков Хаара. Они необходимы для того, чтобы с их помощью можно было распознать лицо. Учёными было доказано, что область человеческого лица в районе глаз темнее области в районе щёк, это означает что если будет подобран такой признак Хаара, что его верхняя область будет темнее нижней на некоторое число, которое больше порогового значения, то можно утверждать, что в этой области содержится лицо. Пороговое значение вычисляется в ходе

обучения классификатора. Для того, чтобы можно было вычислить какая область более темная или более светлая применяется интегральное представление изображения.

Интегральное представление изображения – это матрица, совпадающая по размерам с исходным изображением. В каждом её элементе хранится сумма интенсивностей всех пикселей, находящихся левее и выше данного элемента. Для упрощения данных вычислений необходимо сначала преобразовать изображение в оттенки серого.

Для понимания в чём заключается суть данного представления изображения ниже будет приведён рисунок 7.

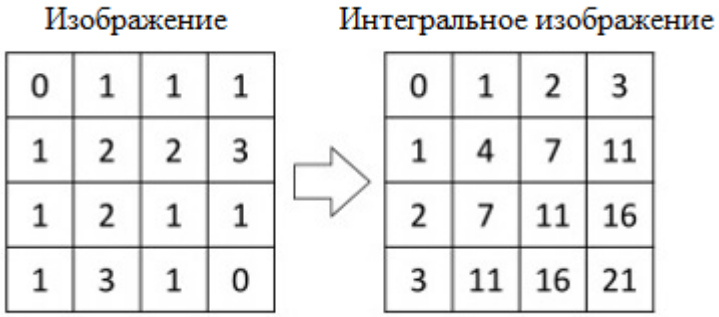

Рисунок 7 – Отличие обычного изображения от интегрального

На рисунке, представленном выше в левой его части пример обычного изображения в оттенках серого, числа – это значения интенсивностей пикселей. В правой части рисунка 7 можно видеть интегральное представление изображение, составленное по указанному выше правилу. С помощью данного представления можно легко вычислить, интенсивность пикселей в любой части исходного изображения и тем самым узнать какая область светлее, а какая темнее.

Преимуществами данного алгоритма являются:

– работает почти без задержек в режиме реального времени даже на процессоре;

– простая архитектура;

– распознаёт лица при разных масштабах изображения.

Недостатками данного алгоритма являются:

– распознавание лица там, где его на самом деле нет;

– не работает на не фронтальных изображениях;

– не работает при окклюзии.

Следующим алгоритмом для распознавания лиц является обратная свёрточная нейронная сеть (DNN). Перед тем как разобраться что же такое DNN необходимо понять, что такое свёрточная нейронная сеть (CNN).

Свёрточная нейронная сеть (CNN) – специальная архитектура искусственных нейронных сетей (ANN), которую предложил в 1988 году французский учёный Yann LeCun, предназначенная для эффективного распознавания образов, входит в состав технологии «глубокого обучения». Данный вид нейронных сетей был разработан опираясь на особенности зрительной коры головного мозга, в ходе её исследования было установлено что нейроны по-разному реагируют на воспринимаемый объект, так некоторые нейроны реагируют на горизонтальные очертания объекта, другие реагируют на вертикальные и так далее. Как любая другая нейронная сеть CNN состоит из нейронных слоёв. Свёрточная нейронная сеть (CNN) может содержать в себе следующие слои:

– свёрточный (convolution),

– подвыборки (pooling),

– активационный (flatten),

– полносвязный (fully connected).

Далее будет подробно рассмотрен каждый слой, но перед этим необходимо ввести несколько ключевых понятий.

Тензор – это 3D массив чисел или говоря другими словами массив матриц чисел. Как известно большинство цифровых изображений хранится в цветовом пространстве RGB, что подразумевает под собой для одного пикселя три числовых компоненты отвечающих за интенсивности красного,

зелёного и синего цветов соответственно. Говоря другими словами все изображения цветового пространства RGB являются тензорами.

Признак – это тензор, который на первых слоях CNN представляет собой какой-либо примитив (горизонтальная линия, кривая линия и т.д.) необходимый для классификации. На более поздних нейронных слоях признаки могут представлять более сложные объекты (уши или глаза).

На рисунке 8 будет представлен пример простейшего признака.

0	0	0	0	0	30	0
0	0	0	0	30	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	30	0	0	0
0	0	0	0	0	0	0

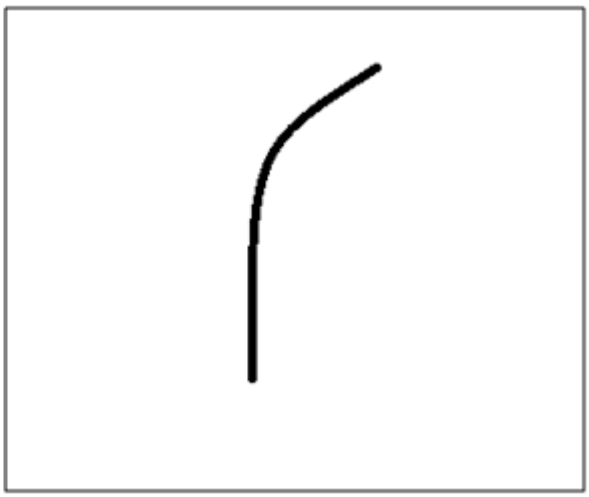

Рисунок 8 – Пример простейшего признака

На данном рисунке изображён простейший признак, то как его видит компьютер и то как его видит человек. Стоит отметить, что данный признак представлен в оттенках серого, поэтому у него всего одна числовая компонента интенсивности цвета.

Карта признаков – это тензор, содержащий в себе результаты преобразований исходного изображения и признаков. Количество каналов в ней зависит от числа используемых признаков. Далее можно переходить к рассмотрению первого из слоёв свёрточной нейронной сети.

Свёрточные слои являются самыми важными в CNN. Как правило первый слой CNN всегда является свёрточным. Задачей данных слоёв является выделение признаков и составление из них карты признаков с помощью, которой в дальнейшем можно классифицировать объект. Для того

чтобы данный слой мог выделять признаки в нём присутствуют так называемые фильтры. Фильтр – это тензор, необходимый для выделения признака, он имеет столько же каналов сколько и входное изображение [6]. Для формирования карты признаков из входного изображения требуется произвести операцию свёртки входного тензора с каждым из признаков. Свёртка – это операция вычисления нового значения заданного пикселя с учётом значений окружающих пикселей [9]. Алгоритм её работы можно представить следующим образом: фильтр устанавливается в левую верхнюю часть входного тензора и производится поэлементное умножение значений фильтра и входного тензора, после чего значения со всех каналов суммируются между собой и со значением смещения данного фильтра [19]. Затем фильтр смещается и данные операции повторяются до того момента пока фильтр не пройдёт весь входной тензор [2]-[3].

На рисунке 9 будет представлен описанный выше процесс формирования признака.

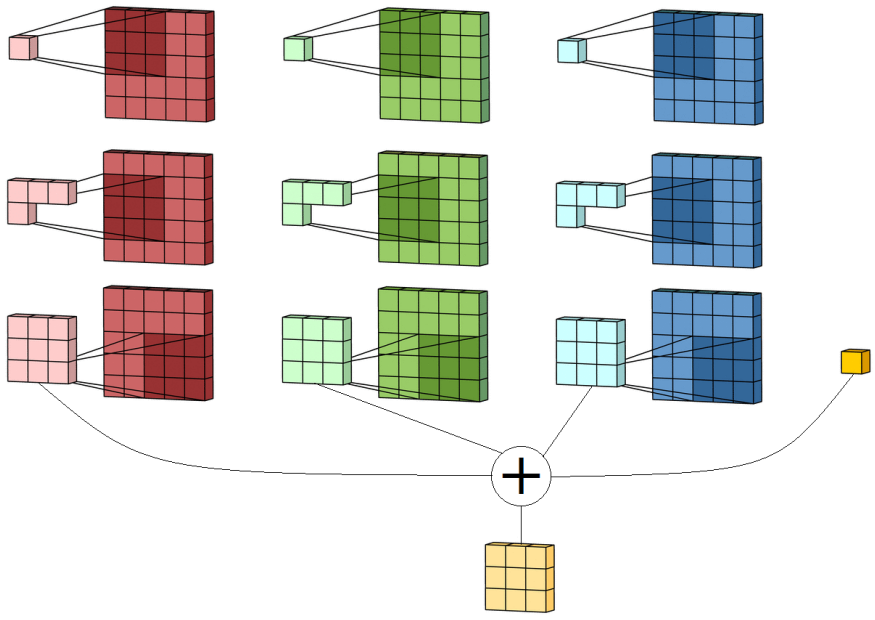

Рисунок 9 – Пример работы свёрточного слоя

На рисунке 9 продемонстрирован ранее описанный процесс. Как можно видеть из данного рисунка число пикселей выходного тензора уменьшилось,

иногда данная ситуация нежелательна для этого входной тензор дополняют нулями тем самым увеличивая его, но не изменяя само изображение, что позволяет при свёртке получить тензор такого же или даже большего размера чем входной. На выходе свёрточного слоя получается карта признаков, которая содержит в себе признаки более высокого уровня и соответственно к ней могут применяться более высокоуровневые фильтры.

Ещё одним слоем свёрточной нейронной сети является слой подвыборки. Задача данного слоя уменьшить признаковое пространство при этом сохранив наиболее важные признаки. Существует 3 разных вида слоя подвыборки:

– максимальный пулинг (max pooling), вычисляется максимальное значение из области подвыборки;

– средний пулинг (average pooling), вычисляется среднее значение из области подвыборки;

– пулинг суммы (sum pooling), вычисляется сумма области подвыборки.

На рисунке 10 будет представлен пример работы слоя подвыборки.

Рисунок 10 – Пример работы слоя подвыборки

На данном рисунке наглядно продемонстрировано то как работают разные виды слоёв подвыборки.

Активационный слой предназначен для нормализации данных, полученных с предыдущих слоёв, с помощью функций активации. Наиболее распространёнными функциями активации являются следующие:

– ReLU,
– leaky ReLU,
– sigmoid,
– tanh.

У каждой из данных функций есть свои плюсы и минусы, но наиболее распространённой для CNN функцией активации является ReLU, в некоторых библиотеках она уже встроена в свёрточный слой.

Последнем слоем свёрточной нейронной сети (CNN) является полносвязный, данный слой ничем не отличается от скрытого слоя в нейронной сети прямого распространения. Все его нейроны связаны со всеми нейронами предыдущего слоя, также они могут содержать вектор смещений. В CNN данный слой чаще всего применяется для классификации объекта, представленного на исходном изображении, полносвязный слой принимает высокоуровневые признаки, превращая их в один большой вектор после чего данный вектор обрабатывается в нейронах этого слоя и на выходе получается вектор значений, количество которых равно количеству классов анализируемого объекта. Однако данный слой не всегда используется как выходной, иногда он также используется и как промежуточный, в таком случае количество нейронов никак не связано с количеством классов объекта. После того как было рассмотрено что такое CNN можно перейти к описанию DNN.

Обратная свёрточная нейронная сеть (DNN) – это CNN работающая в обратном порядке, её разработал Matthew Zeiler для анализа качества распознавания образов при помощи CNN [4]. DNN анализирует фильтры и признаки для выявления частей, а иногда и структур исходного изображения [19]. В ней применяются операции обратного ReLU и обратной свёртки в противовес соответствующим операциям из CNN [10]-[11].

Для большей наглядности и понимания работы обратной свёрточной нейронной сети (DNN) будет представлен рисунок 11 на котором DNN применялась к разным нейронным слоям.

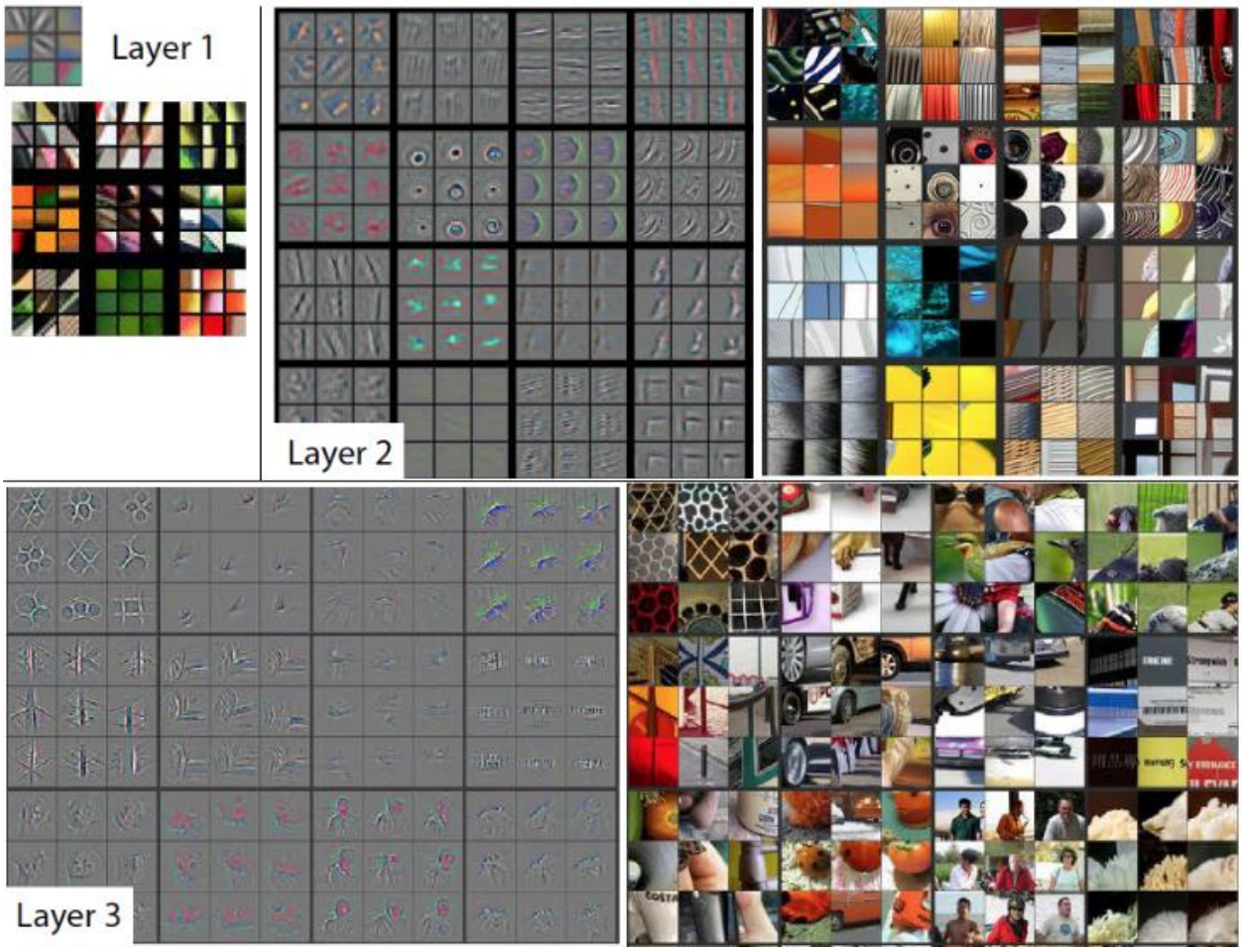

Рисунок 11 – Пример работы DNN

На рисунке, расположенном выше можно видеть, как DNN отображает фильтры, которые были активированы соответствующими кусками цветных изображений. Так в слое 1 (Layer 1) можно видеть, что фильтры реагируют на примитивы цветных изображений, в слое 2 (Layer 2) с увеличением уровня признаков фильтры начинают реагировать на более сложные куски изображения, а в слое 3 (Layer 3) можно видеть, что фильтры могут распознавать человека на фотографии. Нельзя не заметить, что чем больше номер слоя CNN, тем более высокоуровневые признаки на нём встречаются и тем более сложный фильтр активируется. Также стоит отметить, что

обратная свёрточная нейронная сеть (DNN) очень хорошо справляется с определением структур объекта, который активирует то или иной фильтр. Именно благодаря этому DNN очень удобно использовать для определения области в которой располагается лицо человека на изображении. Процесс работы DNN можно описать следующим образом:

– запуск прямого прохода CNN;

– выбор интересующего слоя;

– фиксация активации одного или нескольких нейронов и обнуление остальных;

– запуск обратного вывода данного слоя.

В данной работе будет рассматриваться реализация DNN из библиотеки OpenCV.

Преимуществами данного алгоритма являются:

– самый точный из рассматриваемых алгоритмов;

– работает в режиме реального времени на процессоре;

– распознаёт лица при их повороте вверх, вниз, влево, вправо и так далее;

– работает даже при значительной окклюзии;

– распознаёт лица в различных масштабах.

Недостатков данный алгоритм не имеет, за исключением того, что он работает медленнее алгоритма HOG.

Перед рассмотрением двух оставшихся алгоритмов стоит сказать, что они являются менее распространёнными, чем те что уже были рассмотрены поэтому их описание будет поверхностным.

Гистограмма направленных градиентов (HOG) – это дескриптор, который анализируя изображение определяет, как на нём изменяется яркость. После этого результаты его работы подаются в классификатор SVM, который вычисляет координаты области, в которой располагается искомый объект. В данной выпускной квалификационной работе будет рассматриваться реализация данного алгоритма из библиотеки Dlib.

Преимуществами данного алгоритма являются:

– самый быстрый из рассматриваемых алгоритмов при работе на процессоре;

– максимально точно распознаёт лица, расположенные в анфас;

– менее требовательный к вычислительным ресурсам по сравнению с другими алгоритмами;

– работает при незначительной окклюзии.

Недостатками данного алгоритма являются:

– не распознаёт лица размером меньше чем 80 на 80 пикселей;

– при распознавании в результирующую область не попадает лоб, а иногда и подбородок;

– не распознаёт лица при отклонении положения глаз от фронтального.

Последним рассматриваемым алгоритмом является детектор объектов с максимальным запасом (MMOD). Данный алгоритм является ещё одним представителем «глубокого обучения» и основан на использовании некоторых свойств свёрточной нейронной сети. Рассматриваемая реализация также является частью библиотеки Dlib.

Преимуществами данного алгоритма являются:

– распознаёт лица в различных положениях;

– устойчив к окклюзии;

– работает очень быстро на GPU;

– быстрый тренировочный процесс.

Недостатками данного алгоритма являются:

– очень медленный при работе на процессоре;

– не распознаёт лица размером меньше чем 80 на 80 пикселей;

– результирующая область ещё меньше чем у алгоритма HOG.

Далее будет произведено сравнение ранее описанных алгоритмов.

По точности наивысший показатель при тестировании разного рода изображениями показал алгоритм DNN, второе место занял каскад классификаторов Хаара, а последнее место MMOD.

Тестирование скорости обработки кадра показало, что HOG является наибыстрейшим из рассматриваемых алгоритмов, второе место занял каскад классификаторов Хаара, а последнее MMOD. Стоит отметить, что данное исследование производилось на CPU, так как реализации алгоритмов Виолы-Джонса, DNN и HOG не поддерживают работу на GPU. Поэтому на графическом процессоре видеокарты может работать только детектор объектов с максимальным запасом (MMOD) и при такой работе он несомненно является быстрейшим.

Тестирование на обработку окклюзии показало, что при её небольших значениях лица распознают только DNN и MMOD. При увеличении окклюзии алгоритм MMOD также перестал справляться со своей задачей, но DNN продолжает работать вплоть до перекрытия половины лица.

Однако самым важным моментом является размер результирующей области в которой располагается лицо. Так как распознавание лиц является лишь шагом к распознаванию эмоций то необходимо чтобы в результирующую область было включено абсолютно всё лицо. Однако, как говорилось ранее алгоритмы HOG и MMOD, не захватывают часть лица, поэтому не смотря на некоторые их преимущества они не могут применяться к задаче классификации эмоций. Для подтверждения выше сказанного, а также наглядной демонстрации того как работают данные алгоритмы будет приведён рисунок 12.

На рисунке 12, расположенном ниже будет представлен пример работы описанных ранее алгоритмов.

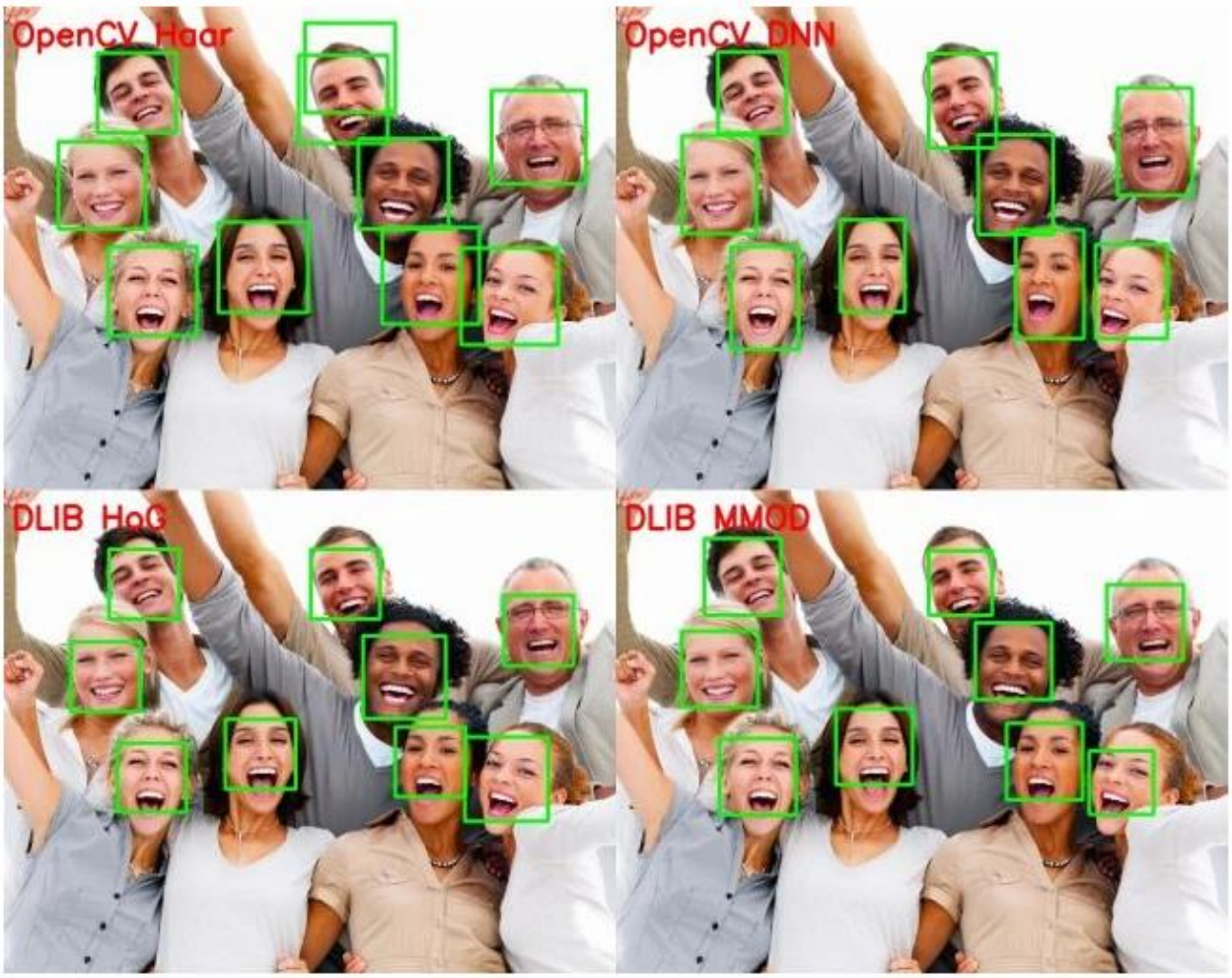

Рисунок 12 – Пример работы распознавателей лиц

На рисунке, расположенном выше можно видеть, то как ранее описанные алгоритмы справляются с задачей поиска лиц на изображении. Можно заметить, что алгоритмы HOG и MMOD, как и говорилось ранее действительно обрезают результирующую область.

В связи с выше сказанным остаются только 2 алгоритма, которые можно применить для распознавания лиц: каскад классификаторов Хаара и обратная свёрточная нейронная сеть (DNN). Поскольку целью является повышение точности распознавания лиц то в качестве распознавателя лиц

будет выбран алгоритм DNN, который по этому показателю намного превосходит все остальные.

После того как был выбран и описан распознаватель лиц, можно переходить ко второй наиболее важной задаче – выбору классификатора эмоций. Данная задача намного проще выбора распознавателя лиц, поскольку наиболее надёжным и точным классификатором эмоций является свёрточная нейронная сеть (CNN). Именно она и будет являться классификатором эмоций в разрабатываемом алгоритме, а так как процесс функционирования CNN был рассмотрен ранее, повторно это делать не требуется. Необходимо отметить что в качестве архитектуры как CNN, так и DNN была выбрана LeNet5 схема которой будет представлена на рисунке 13.

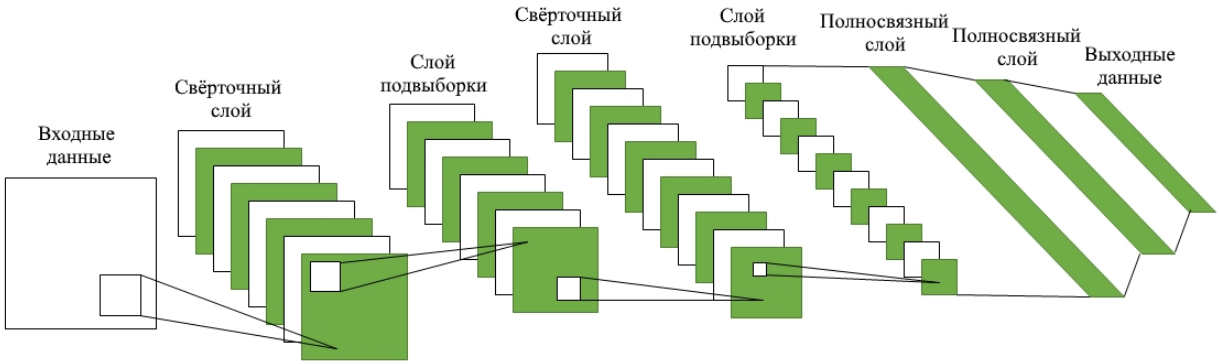

Рисунок 13 – Схема свёрточной нейронной сети LeNet5

На рисунке 13 наглядно представлена структура используемых свёрточной нейронной сети (CNN) и обратной свёрточной нейронной сети (DNN). Данные сети включает в себя 2 свёрточных слоя с функциями активации ReLU, 2 слоя подвыборки с максимальным пулингом и 2 полносвязных слоя. Стоит отметить, что выходной слой сети не является полносвязным в полной мере, он связан с предыдущим слоем соединением Гаусса. Данный вид соединения применяется на выходных слоях CNN и позволяет при помощи евклидовых радиальных базисных функций установить связь между входными данными и их принадлежностями какому-либо классу.

Таким образом в качестве распознавателя лиц была выбрана обратная свёрточная нейронная сеть (DNN), а в качестве классификатора эмоций свёрточная нейронная сеть (CNN). Далее на рисунке 14 будет представлена блок-схема алгоритма распознавания эмоций на изображении.

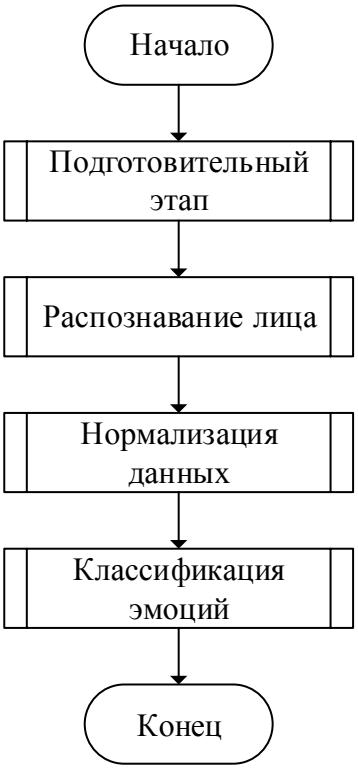

Рисунок 14 – Блок-схема алгоритма распознавания эмоций на изображении

На рисунке, представленном выше можно видеть, что процесс работы алгоритма распознавания эмоций сводится к последовательному запуску четырёх подпроцессов. В первом из них, кадры с веб-камеры, видеофайла или фотографий считываются, их размер преобразуется к размеру кадров на котором обучалась DNN, после чего происходит процесс среднего вычитания необходимый для борьбы с изменениями освещённости кадров и более точной работы обратной свёрточной нейронной сети (DNN). Во втором подпроцессе отрабатывает алгоритм распознавателя лиц DNN. В третьем если было обнаружено несколько лиц, то происходит выделение наибольшей по площади области, в которой обнаружено одно лицо, в качестве отдельного кадра, перевод его в оттенки серого, изменения размера кадра до того на

котором обучалась CNN, а также нормализация интенсивностей цвета пикселей путём деления их значений на значение максимальной интенсивности цвета пикселя. В последнем подпроцессе работает классификатор эмоций алгоритм CNN.

После наглядного представления и пояснения процесса работы алгоритма можно переходить к проектированию его математической модели.

2.2 Математическая модель алгоритма

Сначала будут спроектированы математические модели CNN и DNN, которые в последующем будут дополнены другими формулами. Это необходимо для более лаконичного и понятного изложения информации. Как и в случае с описанием обратной свёрточной нейронной сети (DNN) сначала будет спроектирована математическая модель свёрточной нейронной сети (CNN), которая в дальнейшем будет дополнена формулами до соответствия DNN.

Далее будет рассмотрено проектирование математической модели свёрточного слоя.

На свёрточном слое главной операцией является свёртка, которую можно представить формулой 1:

$$\mathbb{w}_{m,n}^{l} = \{w_{m,n}^{l}(i,j)\}, \quad (1)$$

где $\mathbb{w}_{m,n}^{l}$ – свёртка, применяемая к карте признаков m слоя (l-1), на слое

l с картой признаков n.

Таким образом на выходе свёрточного слоя l, карта признаков n будет вычисляться по формуле 2.

$$y_n^l = f_l(\sum_{m \in V_n^l} y_m^{l-1} \otimes \mathbb{w}_{m,n}^{l} + b_n^l), \quad (2)$$

где y_n^l – n-ая карта признаков на слое l;

f_l – функция активации слоя l;

V_n^l – список всех уровней слоя (l-1), которые соединяются с картой признаков n слоя l;

b_n^l – смещения, присоединяемые к карте признаков n на слое l.

Можно видеть, что в формуле 2 присутствует функция активации, которой является ReLU. Данная функция вычисляется по формуле 3.

$$f_l(x) = \max(0, x) \tag{3}$$

Для того, чтобы вычислить размер выходной карты признаков можно воспользоваться формулой 4.

$$len^l = (H^{l-1} - r^l + 1) \cdot (W^{l-1} - c^l + 1), \tag{4}$$

где len^l – длина карты признаков слоя l;

H^{l-1} – высота входной карты признаков на слое (l-1);

r^l – высота применяемой свёртки на слое l;

W^{l-1} – ширина входной карты признаков на слое (l-1);

c^l – ширина применяемой свёртки на слое l.

На этом проектирование математической модели свёрточного слоя завершено. Далее будет рассмотрено проектирование математической модели слоя подвыборки.

Как известно задача слоя подвыборки уменьшить признаковое пространство, данный процесс можно описать формулой 5.

$$z_n^{l-1} = \max(y_n^{l-1}(2i-1,2j-1), y_n^{l-1}(2i-1,2j), y_n^{l-1}(2i,2j--1),\ y_n^{l-1}), \tag{5}$$

где z_n^{l-1} – значение максимального элемента из области подвыборки;

y_n^{l-1} – n-ая карта признаков (l-1) слоя.

Стоит отметить, что максимальный пулинг является наиболее эффективным по результатам многочисленных исследований.

Далее полученные элементы объединяются в матрицу подвыборки, как показано на формуле 6.

$$\mathbb{z}_n^{l-1} = \{z_n^{l-1}(i,j)\} \tag{6}$$

Таким образом на выходе слоя подвыборки n-ая карта признаков слоя l рассчитывается по формуле 7.

$$y_n^l = \mathbb{z}_n^{l-1} \times \mathbb{w}_n^l + b_n^l \tag{7}$$

Размерности выходной карты признаков можно рассчитать по формулам 8 и 9.

$$H^l = \frac{H^{l-1}}{2}, \tag{8}$$

$$W^l = \frac{W^{l-1}}{2}. \tag{9}$$

На этом проектирование математической модели слоя подвыборки окончено. Поскольку слои активации уже встроен в остальные слои, то его проектирование не требуется. Поэтому далее будет описана математическая модель полносвязного слоя. Его математическая модель будет такой же, как и у скрытого нейронного слоя нейронной сети прямого распространения.

Значения n-го выходного нейрона рассчитываются по формуле 10.

$$y_n^l = f_l(\sum_{m=1}^{N^{l-1}} y_m^{l-1} \times \mathbb{w}_{m,n}^l + b_n^l), \tag{10}$$

где N^{l-1} – количество нейронов на (l-1) слое.

Стоит отметить, что на выходном слое свёрточной нейронной сети (CNN) функцией активации является не ReLU, а sigmoid, что позволяет зафиксировать диапазон возможных значений на интервале от 0 до 1. Данная функция рассчитывается по формуле 11.

$$f_l(x) = \frac{1}{1 + e^{-x}} \tag{11}$$

По окончании работы свёрточной нейронной сети будет получен вектор, количество элементов которого равно количеству классов, его можно описать при помощи формулы 12.

$$\mathbb{y} = [y_1^L, y_2^L, \dots, y_{N^L}^L], \tag{12}$$

где L – номер последнего слоя нейронной сети.

На этом проектирование математической модели свёрточной нейронной сети (CNN) закончено.

Как говорилось ранее обратная свёрточная нейронная сеть (DNN) отличается от CNN тем, что она работает в обратном порядке и использует обратные операции такие как обратный ReLU и обратная свёртка. Далее данные операции будут представлены в виде формул.

Функция активации ReLU, преобразует все значения в неотрицательные, поэтому при обратном проходе через нелинейность также должны использоваться неотрицательные значения. Для этого используется тот же самый ReLU, поэтому формула 3 справедлива для обратного ReLU.

Для выполнения операции обратной свёртки необходимо применить транспонированные фильтры к выходам текущего слоя из обратного ReLU. Значит карта признаков n слоя l при обратной свёртке может быть вычислена по формуле 13.

$$x_n^l = \sum_{m \in V_n^l} f_l(y_m^l) \otimes (\mathbb{W}_{m,n}^l)^T \quad (13)$$

Таким образом при помощи дополнения математической модели CNN, формулой 13, была спроектирована математическая модель DNN.

Далее будет спроектирована полная математическая модель алгоритма.

На вход алгоритму с видеокамеры или из файла подаётся кадр, который представляет собой тензор, который можно представить с помощью формулы 14.

$$cadr_n(i,j) = \{pix_n(i,j)\}, \quad (14)$$

где n – номер цветового канала.

Затем изменяется размер кадра по методу ближайшего соседа, значение нового неизвестного пикселя равняется ближайшему значению старого известного пикселя. Данный процесс можно представить формулами 15-17.

$$k_1 = \frac{H_{old}}{H_{new}}, \quad (15)$$

где k_1 – коэффициент масштабирования по высоте изображения.

$$k_2 = \frac{W_{old}}{W_{new}}, \quad (16)$$

где k_2 – коэффициент масштабирования по ширине изображения.

$$cadr\_resize(i,j) = cadr(k_1 \cdot i, k_2 \cdot j) \quad (17)$$

Затем к изображению применяется процесс среднего вычитания в ходе которого изменяются интенсивности цвета пикселей всех его каналов. Стоит

отметить, что для каждого канала применяется одно и тоже значение для вычитания. Описанный ранее процесс можно представить формулой 18.

$$cadr\_norm_n(i,j) = cadr\_resize_n(i,j) - sredn_n \quad (18)$$

После этого кадр обрабатывается алгоритмом DNN, который функционирует по формулам 1-13. На выходе DNN получается массив областей, в которых расположены лица, его можно описать формулой 19.

$$lica(i) = \{x_0(i), y_0(i), W(i), H(i)\}, \quad (19)$$

Затем из данного массива выбирается одна область наибольшая по площади, данный процесс можно представить формулой 20.

$$lico\_obl = lica(\underset{el \in lica}{argmax}(el(2) \cdot el(3))) \quad (20)$$

После этого исходный кадр с изменённым размером переводится в оттенки серого по формуле 21.

$$cadr\_gray = cadr\_resize_0 \times 0{,}299 + cadr\_resize_1 \times \\ \times 0{,}587 + cad\_resize_2 \times 0{,}114 \quad (21)$$

Далее выбранная область копируется в новый массив по формуле 22.

$$lico(i,j) = cadr\_gray(k,l) \quad (22)$$

Затем по формулам 15-17 изменяется размер кадра с лицом и записывается в массив lico_resize. После этого происходит нормализация данных на отрезке от 0 до 1, по формуле 23.

$$lico\_norm(i,j) = \frac{lico\_resize_{(i,j)}}{255}, \quad (23)$$

Далее нормированные данные подаются в алгоритм CNN, который работает по формулам 1-12 и в результате его работы вычисляются критерии силы каждой эмоции.

В данном разделе были рассмотрены различные виды распознавателей лиц, был произведён их сравнительный анализ, по результатам которого в качестве применяемого распознавателя лица в данном алгоритме была выбрана DNN. Также было решено использовать в качестве классификатора эмоций CNN. Затем была спроектирована математическая модель алгоритма.

3 ПРАКТИЧЕСКАЯ РЕАЛИЗАЦИЯ ПРЕДЛОЖЕННЫХ РЕШЕНИЙ И ТЕСТИРОВАНИЕ

3.1 Программная реализация алгоритма

Спроектированный ранее алгоритм будет реализован на языке программирования Python версии 3. Для работы с изображениями, видеопотоками и для запуска ранее спроектированной обратной свёрточной нейронной сети (DNN) будет использоваться библиотека OpenCV. Для того, чтобы была возможность запускать свёрточную нейронную сеть (CNN) для классификации эмоций будет задействована программная библиотека для машинного обучения TensorFlow, для неё будет использоваться надстройка Keras. Также для того, чтобы обработка изображений при классификации эмоций происходила не на процессоре, а на видеокарте будет использована технология CUDA.

Python – язык программирования высокого уровня общего назначения, благодаря своему достаточно простому синтаксису он является очень популярным [20]. Его разработал Guido van Rossum в 1989 году. Python поддерживает структурное, объектно-ориентированное, функциональное, императивное и аспектно-ориентированное программирование, является активно развивающемся языком программирования, обновления с различными нововведениями выходят примерно раз в 2,5 года. В настоящее время существуют две отличающихся между собой по синтаксису версии: вторая и третья [14]-[15]. Так как к моменту написания данной выпускной квалификационной работы Python 2 считается устаревшим, и его поддержка в 2020 году прекратилась, то выбор был сделан в пользу 3 версии.

Преимуществами Python 3 являются:

– данный язык является интерпретируемым, что значительно упрощает отладку написанной на нём программы;

– динамическая типизация, что означает отсутствие явного определения типа переменной при её объявлении, в процессе работы программы ей будет назначен тип в соответствии с её значением;

– автоматический сборщик мусора, исключающий возможность утечки памяти;

– интеграция языков C и C++, для расширения базовых возможностей;

– лаконичный и простой синтаксис, который способствует ясному отображению кода;

– большое множество уже готовых программных модулей и библиотек, которые позволяют достаточно просто решать даже самые сложные задачи;

– кроссплатформенность, что означает поддержку программы, написанной в одной операционной системе, а запускаемой в любой другой операционной системе.

Недостаток же Python 3 связан с одним из его преимуществ, он является интерпретируемым языком программирования, что означает сниженную скорость выполнения программы.

OpenCV – это библиотека компьютерного зрения с открытым исходным кодом, в которой содержатся алгоритмы компьютерного зрения, обработки изображение и другие полезные функции. Данная библиотека была разработана компанией Itseez, которая была куплена компанией Intel в 2016 году. OpenCV была реализована для C и C++, но в данный момент времени разрабатывается и для Python, Java, Ruby, а также других языков программирования. Данная библиотека необходима для того, чтобы получать изображения в том виде в котором они удобны для анализа и обработки. В случае работы с видеопотоком она позволяет разбить его на кадры и обрабатывать их как обычные изображения.

TensorFlow – это открытая программная библиотека для машинного обучения, разработанная компанией Google. Данная библиотека включает в себя необходимые методы и алгоритмы для запуска и обучения нейронных сетей предназначенных для распознавания объектов. Стоит отметь, что нейронные сети, обученные с помощью алгоритмов из TensorFlow в области

распознавания объектов, могут не уступать человеку. Основной API для работы с библиотекой реализован для Python [17]-[18].

Keras – это открытая библиотека, написанная на языке Python. Основным её разработчиком является инженер Google François Chollet. В Python данная библиотека является надстройкой над TensorFlow, то есть она предоставляет высокоуровневый, более интуитивный набор абстракций, который делает простым формирование нейронных сетей, независимо от используемой в качестве бэкенда библиотеки [13].

CUDA – это платформа параллельных вычислений и интерфейс прикладного программирования (API), созданная компанией «Nvidia». Данная платформа помогает разработчикам программного обеспечения увеличить вычислительную производительность используя графические процессоры фирмы «Nvidia». Платформа CUDA – программный уровень, который предоставляет прямой доступ к виртуальному набору команд GPU и вычислительным ядрам. Библиотеки CUDA с GPU-ускорением обеспечивают ускорение вставки во многих областях, таких как линейная алгебра, обработка изображений и видео, «глубокое обучение» и аналитика графов. Приложения на базе CUDA могут быть развернуты во всех семействах графических процессоров «NVIDIA», доступных в домах и на стендах графических процессоров в облаке. Используя встроенные возможности для распределения вычислений по конфигурациям с несколькими графическими процессорами, ученые и исследователи могут разрабатывать приложения, которые масштабируются от рабочих станций с одним графическим процессором до облачных установок с тысячами графических процессоров. Данная технология может применяться в таких языках программирования как C, C++ и Fortran, также существую функции, ускоренные при помощи CUDA, которые можно применять и в языке программирования Python. Для того, чтобы было возможно применить технологию CUDA необходимо чтобы в ПК была установлена видеокарта с графическим чипом от компании NVIDIA, также должно быть установлено соответствующее программное

обеспечение, которое можно найти на официальном сайте компании. Так как в компьютере, на котором ведётся разработка данного приложения, установлена видеокарта Nvidia Geforce GTX 980, технологию CUDA применить возможно. Стоит отметить, что программная реализация алгоритма будет работать и без технологии CUDA, однако её использование позволит значительно сократить время классификации эмоций, так как процессор может одновременно обработать столько потоков сколько у него ядер (или в 2 раза больше зависит от архитектуры), а видеокарта способно обработать сотни тысяч потоков единовременно.

Далее будут приведены важнейшие фрагменты кода программной реализации данного алгоритма для наглядного представления и понимания как он будет работать.

На рисунке, представленном ниже будет продемонстрирован фрагмент кода с импортами различных необходимых библиотек и программных модулей.

```
from tkinter import *
from tkinter import filedialog as fd
from tkinter import messagebox as mb
from PIL import ImageFont, ImageDraw, Image, ImageTk
from keras.preprocessing.image import img_to_array
from keras.models import load_model
import cv2
from time import time
import numpy as np
import pandas as pd
```

Рисунок 15 – Фрагмент кода с импортом библиотек и модулей

На рисунке 15, можно видеть, как к приложению подключаются все необходимые для его работы библиотеки.

Tkinter – это графическая библиотека, благодаря которой можно создавать программы с оконным интерфейсом. К тому же в ней содержатся необходимые шаблоны, для создания диалоговых окон выбора файла, а также информационные окна для диалога с пользователем.

PIL – это библиотека необходимая для различных манипуляций с изображениями. В данном приложении применяется для вставки изображений на форму, а поскольку в реализации OpenCV для Python отсутствует возможность вывода кириллицы на изображение также необходима для отображения русского текста на нём.

Про библиотеки Keras и использующуюся в качестве бэкенда TensorFlow было рассказано ранее также, как и про реализацию для Python библиотеки OpenCV, которая называется cv2.

Time – это программный модуль необходимый для работы со временем. В данное приложение подключён для синхронизации потоков при работе с веб-камерой.

Numpy – это библиотека, позволяющая работать с многомерными массивами и матрицами, также предоставляет доступ к высокоуровневым и очень эффективным функциям для обработки данных массивов и матриц. В данном приложении необходима для представления изображения в виде многомерной матрицы, которая в дальнейшем обрабатывается нейронными сетями [12].

Pandas – это библиотека содержащая в себе необходимый высокоуровневый функционал необходимый для анализа данных. В данном приложении применяется в ходе тестирования для извлечения из базы изображений тестовых данных [16].

Также стоит отметить, что обучение обратной свёрточной нейронной сети (DNN) для распознавания лица и свёрточной нейронной сети (CNN) – классификатора эмоций для повышения их точности требует длительных временных затрат, исчисляемых днями, а также достаточно хорошие вычислительные машины, которые в ходе обучения могут пострадать из-за перегрева. Поэтому для того чтобы избежать указанные ранее ситуации обученные варианты упомянутых выше нейронных сетей были скачаны из интернета.

Далее на рисунке 16 будет представлен процесс подключения моделей нейронных сетей, а также другие необходимые глобальные переменные.

```
fon1X = 5; fon2X = 500; fonY=35
fon1W = 500; fon2W = 335; fonH = 380

modelEmo = 'models/_mini_XCEPTION.102-0.66.hdf5'
klasificatorEmo = load_model(modelEmo, compile=False)
modelLico = "models/opencv_face_detector_uint8.pb"
configModelLico = "models/opencv_face_detector.pbtxt"
net = cv2.dnn.readNetFromTensorflow(modelLico, configModelLico)
EMOTIONS = ["Гнев", "Отвращение", "Страх", "Счастье", "Печаль", "Удивление", "Спокойствие"]
testNabori = ["gnev.csv", "otvrashenie.csv", "strax.csv", "schastie.csv", "pechal.csv", "ydivlenie.csv", "spokoistvie.csv"]
imaShrifta = "19219.ttf"
shrift = ImageFont.truetype(imaShrifta, 14)

camera = cv2.VideoCapture()
video = cv2.VideoCapture()
```

Рисунок 16 – Фрагмент кода с определением глобальных переменных

На данном рисунке можно видеть все глобальные переменные, которые используются в приложении. Первая и вторая строчки рисунка относятся к графической части, то есть определяются положение и размер рабочих областей. На следующей строке с кодом определяется переменная modelEmo, которая будет хранить в себе путь к модели свёрточной нейронной сети (CNN) предназначенной для классификации эмоций. После этого определяется переменная klasificatorEmo, в которую при помощи библиотеки Keras загружается ранее указанная модель, флаг compile=false указывает на то, что файл с моделью компилировать не требуется. Затем определяются переменные modelLico и configModelLico, первая из них содержит путь к самой модели обратной свёрточной нейронной сети (DNN), вторая указывает на путь к конфигурационному файлу данной сети. Далее определяется переменная net, которая является рабочей моделью нейронной сети, она будет построена с помощью библиотеки Keras, на основе модели DNN и её конфигурационного файла. Затем определяется список EMOTIONS, который хранит в себе 6 базовых эмоций, указанных ранее, а также состояние спокойствия. Далее определяется список testNabori, который хранит в себе имена всех файлов, с тестовыми наборами данных. На двух следующих строках представлено определение переменной imaShrifta, хранящей путь к используемому шрифту и определение переменной shrift, хранящей в себе

информацию о шрифте и его размере, они необходимы для вывода кириллицы на изображение. На последних строках определены переменные для работы с видеопотоками.

На рисунке, изображённом ниже представлен фрагмент кода с реализацией алгоритма для распознавания эмоций на изображении.

```
def raspoznavanieEmosii(cadr, emoSredn):
    cadr = cv2.resize(cadr, (300, 300))
    cadrOpencvDnn = cadr.copy()
    cadrH = cadrOpencvDnn.shape[0]
    cadrW = cadrOpencvDnn.shape[1]
    signal = cv2.dnn.blobFromImage(cadrOpencvDnn, 1.0, (300, 300), [104, 117, 123], True)
    net.setInput(signal)
    raspoznanie = net.forward()

    lica = []
    for i in range(raspoznanie.shape[2]):
        doverie = raspoznanie[0, 0, i, 2]
        if doverie > 0.7:
            x1 = int(raspoznanie[0, 0, i, 3] * cadrW)
            y1 = int(raspoznanie[0, 0, i, 4] * cadrH)
            x2 = int(raspoznanie[0, 0, i, 5] * cadrW)
            y2 = int(raspoznanie[0, 0, i, 6] * cadrH)
            if 0 <= x1 and 0 <= x2-x1 and x2 <= cadrW and 0 <= y1 and 0 <= y2-y1 and y2 <= cadrH:
                lico = x1, y1, x2-x1, y2-y1
                lica.append(lico)

    if len(lica) == 0:
        return False, cv2.cvtColor(cadr, cv2.COLOR_BGR2RGB), 0

    fon = np.zeros((250, 300, 3), dtype="uint8")

    lica = sorted(lica, reverse=True, key=lambda x: (x[2] * x[3]))[0]
    (fX, fY, fW, fH) = lica

    cadrSeryi = cv2.cvtColor(cadr, cv2.COLOR_BGR2GRAY)
    oblastCadra = cadrSeryi[fY:fY + fH, fX:fX + fW]
    oblastCadra = cv2.resize(oblastCadra, (64, 64))
    oblastCadra = oblastCadra.astype("float") / 255.0
    oblastCadra = img_to_array(oblastCadra)
    oblastCadra = np.expand_dims(oblastCadra, axis=0)

    preds = klasificatorEmo.predict(oblastCadra)[0]
    idPreoblEmo = preds.argmax()
    preoblEmotion = EMOTIONS[idPreoblEmo]

    idPreoblEmoSredn = 0
    if emoSredn != None:
        emoSredn[idPreoblEmo] += 1
        idPreoblEmoSredn = np.argmax(emoSredn)

    for (i, (emotion, prob)) in enumerate(zip(EMOTIONS, preds)):
        cvet = (0,255,255,0)
        if EMOTIONS[i] == preoblEmotion:
            cvet = (0,255,0,0)
            cadrImg = Image.fromarray(cadr)
            draw = ImageDraw.Draw(cadrImg)
            draw.text((fX, fY - 15),  preoblEmotion, font = shrift, fill = cvet)
            cadr = np.array(cadrImg)
            cv2.rectangle(cadr, (fX, fY), (fX + fW, fY + fH), (0, 0, 255), 2)
        text = "{}: {:.2f}%".format(emotion, prob * 100)
        if emoSredn != None:
            text += " ({:.2f}%)".format(emoSredn[i] * 100 / sum(emoSredn))
            if idPreoblEmoSredn == i:
                cvet = (255,165,0,0)
        fonImg = Image.fromarray(fon)
        draw = ImageDraw.Draw(fonImg)
        draw.text((10, (i * 35) + 23),  text, font = shrift, fill = cvet)
        fon = np.array(fonImg)
    return True, cv2.cvtColor(cadr, cv2.COLOR_BGR2RGB), cv2.cvtColor(fon, cv2.COLOR_BGR2RGB)
```

Рисунок 17 – Фрагмент кода с реализацией алгоритма распознавания эмоций

На рисунке, изображённом выше можно видеть, то как устроен ранее спроектированный алгоритм. В данную функцию подаётся изображение так

называемый кадр, полученный либо с видеокамеры путём разделения видеопотока на части с помощью метода read, либо загружается изображение с компьютера в формате JPG, JPEG или PNG, также в функцию подаётся список emoSredn, который необходим для классификации эмоций на видеопотоке, то есть этот список содержит информацию об изменении эмоционального состояния человека с течением времени. Далее для использования кадра в обратной свёрточной нейронной сети (DNN) необходимо привести его размер к такому на котором обучалась сеть, в данном случае это размер 300 на 300 пикселей. Данный процесс выполняется с помощью функции resize. Затем, чтобы исходный кадр не был изменён, он копируется в переменную, с которой дальше будет работать DNN, с помощью метода copy. Далее запоминаются размеры исходного кадра, его высота и ширина, для этого используется метод shape, который в зависимости от атрибута выводит размерность того или иного измерения массива. Затем происходит изменение кадра перед тем как подать его в DNN. Из интенсивностей пикселей в формате RGB вычитаются средние значение этих компонент, которые подобраны специально для распознавания лиц. Данный процесс осуществляется с помощью функции blobFromImage, в которую подаётся изменяемое изображение, масштаб изображения после изменения, если данный параметр равен 1, то изображение не изменяется, также подаются средние значения вычитания для каждой компоненты в формате RGB и флаг, который отвечает за представление цветового пространства, если он равен True, то цветовое пространство будет изменено на BGR, иначе RGB. Поскольку данная модель DNN требует цветовое пространство BGR то флаг равен True. После изменения кадра результат подаётся на вход DNN, и она распознаёт на изменённом кадре области в которых скорее всего находятся человеческие лица. Для выполнения данных действий используется метод forward. Далее происходит проверка на то действительно ли найденные области содержат лица. В ходе работы DNN вычисляются коэффициенты доверия, которые отображают то насколько

точно сеть уверена в том, что в данной области есть лицо, они сравниваются с заданным коэффициентом доверия в результате чего отбрасываются сомнительные области, а доверенные области записываются в список. По окончании работы данного цикла в список lica будут записаны все области с лицами. Далее идёт проверка на то действительно ли были найдены лица, если нет работа алгоритма заканчивается, исходный кадр не изменяется, иначе работа алгоритма продолжается. Затем создаётся массив fon указанной размерности и заливается чёрным цветом с помощью метода zeros. В данный массив будет записываться информация об эмоциональном состоянии человека. Поскольку на изображении могут быть несколько человек, а было запланировано распознать эмоции только одного, необходимо выбрать эмоции какого лица подлежат классификации. Для этого с помощью сортировки в обратном порядке по ключу, который представляет собой значение площади области лица, выбирается самая большая область лица. Данный процесс реализуется с помощью функции sorted. Для дальнейших вычислений данная область разбивается на компоненты: координаты начальной точки, ширина и высота области. После этого у исходного кадра изменяется цветовое пространство на оттенки серого и сохраняется в новую переменную, данный процесс необходим для того, чтобы ускорить работу свёрточной нейронной сети (CNN), поскольку одну компоненту обрабатывать быстрее чем три. Для выполнения данных действий применяется функция cvtColor. После этого на кадре в оттенках серого выделяется ранее распознанная область лица, затем её размер приводится к размеру 64 на 64 пикселя, так как именно такой размер изображений требуется для работы свёрточной нейронной сети (CNN), после чего происходит нормализация данных и преобразование исходной двумерной матрицы в многомерный массив, так как именно с такими данными работает CNN. Изменение матрицы происходит с помощью функций img_to_array и expand_dims, первая преобразует исходную матрицу в трёхмерный массив, делая каждый пиксель массивом, а вторая добавляет ещё одно измерение,

превращая массив в четырёхмерный. Далее данные подаются в классификатор эмоций, который вычисляет силу эмоций в диапазоне от 0 до 1. После этого начинается процесс вывода полученных данных на исходный кадр и в ранее созданный массив fon. На исходном кадре вокруг лица рисуется распознанная область и подписывается преобладающая эмоция. В массив fon выводятся все эмоции вместе с критериями силы, а преобладающая выделяется особым цветом. Затем при выводе результатов работы данного алгоритма кадр и массив fon, который тоже является изображением, переводятся из цветового пространства OpenCV, то есть BGR, в цветовое пространство RGB.

На рисунке 18, изображённом ниже будет представлен результат работы разработанного алгоритма.

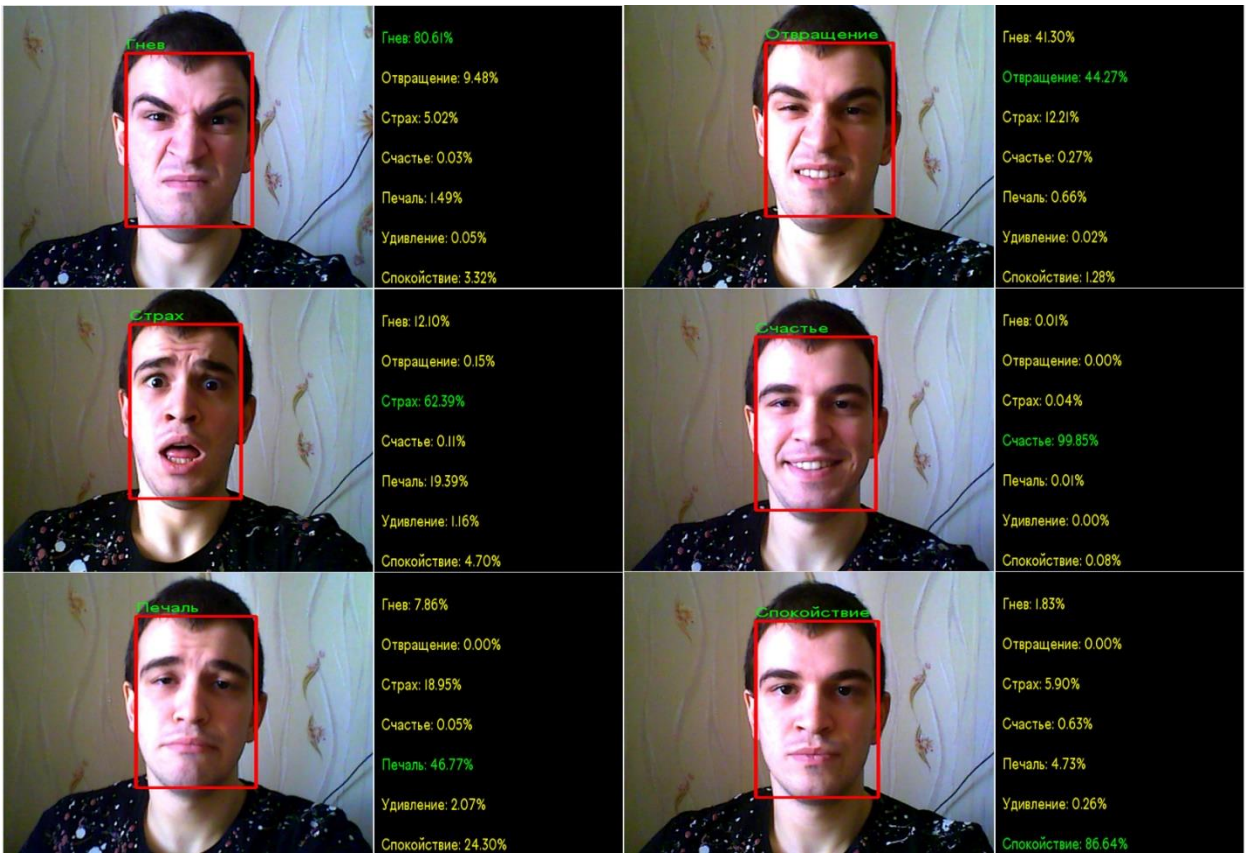

Рисунок 18 – Пример работы разработанного алгоритма Эмоциус

Как можно видеть из рисунка 18, разработанный алгоритм справился с задачей по распознаванию эмоций. Однако неизвестно правильно ли он

сумел распознать преобладающие эмоции, данный момент предстоит выяснить в ходе тестирования алгоритма.

3.2 Тестирование алгоритма

Перед началом тестирования необходимо обозначить его цель. Разработанный тест предназначен для того, чтобы выявить то насколько точно происходит классификация преобладающих эмоций на изображении. Для этого необходимо обладать достаточно большим количеством изображений с лицами людей. Поскольку данные такого рода искать очень долго, были использованы наборы данных, которые содержат в себе до 30 тысяч изображений с различными лицами людей. Об этих наборах следует рассказать подробнее.

FER2013 – набор данных, содержащий в себе более 35 тысяч изображений лиц на них присутствуют все 6 базовых эмоций, а также состояние спокойствия. Размер изображений 48 на 48 пикселей, все они представлены в оттенках серого, так как именно с такими изображениями работает большинство свёрточных нейронных сетей (CNN). Лица на изображениях автоматически были расположены таким образом, чтобы они занимали примерно одинаковое место и располагались по центру. Каждая ячейка данного набора содержит в себе метку, отображающую к какому типу, относится данная эмоция и значения интенсивностей пикселей данного изображения. Так как данные в наборе уже размечены они могут использоваться для обучения и тестирования нейронных сетей. Всего в данном наборе 28709 обучающих примеров, 3589 публичных тестовых примеров и 3589 итоговых тестовых примеров.

Cohn-Kanade – набор данных выпущенный в 2000 году с целью содействия проведению исследований в области автоматического обнаружения индивидуальных черт лица человека. Данный набор предусматривал наличие на изображениях лиц, но не предназначался для тестирования классификаторов эмоций. Поэтому метки эмоций присутствуют, но они не являются достоверными. Размер изображений такой

же, как и в FER2013 и они также представлены в оттенках серого. Так как метки данного набора не являются достоверными использовать его для обучения нейронной сети не следует.

После того как были выбраны наборы данных их необходимо представить в удобном для тестирования виде. Из данных наборов случайным образом были выбраны изображения после чего они были рассортированы по файлам. Стоит отметить что выборка и сортировка изображений производились программным путём.

Далее на рисунке 19 будет представлен фрагмент кода с частью реализации алгоритма тестирования.

```
def zapuskTesta():
    viklCamery()
    viklVideofile()
    zapustitTest.config(state = 'disabled')
    zapustitTest.update()

    itogoEmosiiDano = []
    itogoEmosiiNaideno = [0] * 7
    tochnost = [0] * 7
    itogoEmosiiNaidenoDopTest = [0] * 7
    tochnostDopTest = [0] * 7
    otsechenie = 200
    test = "Основной тест\nКоличество эмоций каждого типа: случайное\n\n"
    dopTest = "Дополнительный тест\nКоличество эмоций каждого типа: " + str(otsechenie) + "\n\n"
    for i in range(len(testNabori)):
        dopTest += "Из эмоций типа \"" + str(EMOTIONS[i]) + "\" было найдено:\n"
        data = pd.read_csv("test/" + testNabori[i])
        izoRazmer = (48,48)
        pixels = data["pixels"].tolist()
        fW, fH = 48, 48
        lica = []
        for pixelsRiad in pixels:
            lico = [int(pixel) for pixel in pixelsRiad.split(' ')]
            lico = np.asarray(lico).reshape(fW, fH)
            lico = cv2.resize(lico.astype("uint8"),izoRazmer)
            lica.append(lico.astype("float32"))
        lica = np.asarray(lica)
        lica = np.expand_dims(lica, -1)

        itogoEmosiiDano.append(lica.shape[0])
        test += "Эмоций типа \"" + str(EMOTIONS[i]) + "\" должно быть: " + str(itogoEmosiiDano[i]) +"\n"

        tekyshEmosia = [0] * 7
        tekyshEmosiaDopTest = [0] * 7
        for j in range(len(lica)):
            idEmosia = raspoznavanieEmosiiTest(lica[j])
            tekyshEmosia[idEmosia] += 1
            if j < otsechenie:
                tekyshEmosiaDopTest[idEmosia] += 1

        test += "Было найдено:\n"
        for j in range(len(tekyshEmosia)):
            itogoEmosiiNaideno[j] += tekyshEmosia[j]
            itogoEmosiiNaidenoDopTest[j] += tekyshEmosiaDopTest[j]
            test += "Эмоций типа \"" + str(EMOTIONS[j]) + "\": " + str(tekyshEmosia[j]) + "\n"
            dopTest += "Эмоций типа \"" + str(EMOTIONS[j]) + "\": " + str(tekyshEmosiaDopTest[j]) + "\n"
        test += "\n"
        dopTest += "\n"
        tochnost[i] = round(tekyshEmosia[i] / itogoEmosiiDano[i] * 100)
        tochnostDopTest[i] = round(tekyshEmosiaDopTest[i] / otsechenie * 100)
    sredn Tochnost = round(sum(tochnost)/len(tochnost))
    sredn TochnostDopTest = round(sum(tochnostDopTest)/len(tochnostDopTest))
```

Рисунок 19 – Фрагмент кода с частью реализации алгоритма тестирования

На рисунке, расположенном выше можно видеть часть реализации алгоритма тестирования, остальной код не был представлен так как в нём реализован вывод результатов тестирования в файл, что не вызывает особого интереса. Из рисунка 19 видно, что для запуска данной функции не требуется передавать ей параметры, для её работы потребуется лишь одна глобальная переменная testNabori о которой говорилось ранее. Перед запуском алгоритма тестирования требуется провести подготовительный этап в ходе которого останавливается чтение видеопотока с камеры (функция viklCamery), останавливается чтение видеопотока из файла (функция viklVideofile), происходят изменения в графическом интерфейсе и определяются все необходимые для тестирования переменные. Так в список itogoEmosiiDano будет записываться общее количество эмоций каждого типа, в itogoEmosiiNaideno будет записываться то сколько эмоций каждого типа было распознано в ходе работы алгоритма, в список tochnost будет помещаться среднее значения точности распознавания для каждой эмоции, itogoEmosiiNaidenoDopTest и tochnostDopTest являются аналогами выше рассмотренных списков, но используются в дополнительном тестировании, переменная otsechenie указывает на то сколько изображений для каждого типа эмоций будет рассматриваться в ходе дополнительного тестирования, а переменные test и dopTest будут содержать в себе результаты основного и дополнительного тестирований соответственно. После подготовительного этапа начинается непосредственно тестирование. В главном цикле последовательно начинают считываться csv файлы, соответствующие рассматриваемым эмоциям, для удобной работы с csv файлами используется функция read_csv, из столбца pixels всех ячеек конкретного файла считываются пиксели, после чего они преобразуются с помощью функции resize в изображение размером 48 на 48 пикселей и добавляются в список lica. После того как все изображения выбранного файла считаны, а lica заполнен, обновляется список itogoEmosiiDano в него добавляется количество только что считанных изображений. Данный процесс осуществляется с помощью

метода append. После этого определяются списки tekyshEmosia и tekyshEmosiaDopTest для того чтобы записывать в них количество распознанных эмоций каждого типа. Тестовый алгоритм распознавания работает также, как и ранее рассмотренный за исключением того, что на всех изображениях присутствуют лица, а значит запуск обратной свёрточной нейронной сети (DNN) для распознавания лиц не требуется областью лица считается всё изображение. После того как были рассмотрены все изображения из выбранного файла происходит подготовка записи информации в файл, а также оценка точности алгоритма на данной эмоции. Точность распознавания эмоции выбранного типа рассчитывается путём деления числа положительных исходов (количество распознаний данной эмоции) на общее число исходов (общее количество изображений из выбранного файла), умножением данного значения на 100 для преобразования в проценты и его округлением. Данный процесс можно представить формулой 24.

$$tochnost_i = round(\frac{tekyshEmosia_i}{itogoEmosiiDano_i} \cdot 100) \quad (24)$$

По данной формуле рассчитывается и точность для дополнительного тестирования. После того как все csv файлы были рассмотрены, а промежуточные точности подсчитаны вычисляется средняя точность работы алгоритма. Она равняется сумме промежуточных точностей, делённой на количество рассматриваемых эмоций, после чего полученное значение округляется. Данные преобразование можно представить в виде формулы 25.

$$srednTochnost = round(\frac{\sum_{i=0}^{n-1} tochnost_i}{n}) \quad (25)$$

Также по данной формуле рассчитывается средняя точность работы алгоритма на дополнительном тесте. После этого работа вычислительной части алгоритма тестирования заканчивается происходит сбор и обработка результирующих данных и вывод их в файл.

На рисунке 20 будет представлена гистограмма с результатами работы алгоритма тестирования.

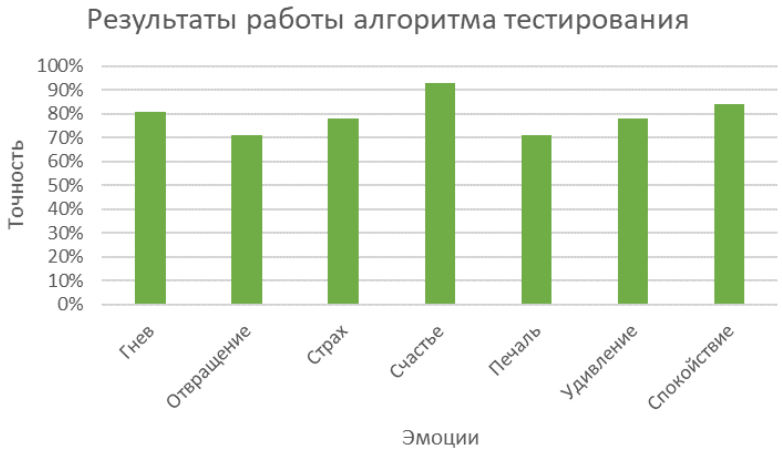

Рисунок 20 – Гистограмма с результатами работы алгоритма тестирования

На рисунке 20, представлена средняя точность алгоритма распознавания эмоций на дополнительном тесте. Из данного рисунка видно, что лучше всего алгоритм распознаёт эмоцию счастье 93% точности, а хуже всего 71% эмоции отвращение и страх. Средняя точность на данном тестировании равняется 79%, что также, как и на основном тесте является очень хорошим показателем.

В данном разделе был произведён выбор языка программирования, применяемого для реализации ранее спроектированного алгоритма, описаны его преимущества и недостатки, рассмотрены различные модули и библиотеки необходимые при реализации алгоритма, после чего был реализован сам алгоритм, а также продемонстрирован результат его работы. Далее был разработан алгоритм тестирования для оценки точности полученного алгоритма распознавания эмоций, после чего были представлены результаты его работы.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной выпускной квалификационной работы был спроектирован и реализован алгоритм для распознавания эмоций на изображении. Данный алгоритм может распознавать все 6 базовых эмоций, а также состояние спокойствия, при работе с видеопотоком алгоритм оценивает не только текущие эмоциональное состояние человека, но и производит расчёты в соответствии с тем какое количество времени объект прибывал в данном эмоциональном состоянии. Повышенная точность распознавания лиц была достигнута использованием для их поиска обратной свёрточной нейронной сети (DNN), достижение точности алгоритма в районе 80% было возможно благодаря классификатору эмоций, которым является свёрточная нейронная сеть (CNN). Считаю, что цель работы была достигнута.

В каждом из основных разделов данной работы решались свои задачи.

В первом разделе была рассмотрена сущность задачи распознавания эмоций, а также представлены наиболее известные существующие аналоги.

Во втором разделе были предложены и рассмотрены решения необходимые при проектировании алгоритма, был сформирован конечный вариант разрабатываемого алгоритма, а также спроектирована его математическая модель.

В третьем разделе были рассмотрены язык программирования, используемый для реализации алгоритма, необходимые библиотеки и технологии, представлены наиболее важные фрагменты кода, а также проведено тестирование точности данного алгоритма.

В дальнейшем разработанный алгоритм можно использовать при проведении социологических исследований или, например, можно встроить его в камеры видеонаблюдения в магазинах, которые буду оценивать эмоциональное состояния покупателей, а результаты выводить на информационное табло. При должном обслуживании такой ход может помочь в привлечении новых клиентов.

СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ

1. Визильтер Ю. В. Обработка и анализ изображений в задачах машинного зрения: Курс лекций и практических занятий. – М.: Физматкнига, 2010. – 672 с.
2. Галушкин, А.И. Нейронные сети: основы теории. / А.И. Галушкин. - М.: РиС, 2015. - 496 с.
3. Гелиг, А. Х. Введение в математическую теорию обучаемых распознающих систем и нейронных сетей. Учебное пособие / А.Х. Гелиг, А.С. Матвеев. - М.: Издательство СПбГУ, 2014. - 224 с.
4. Головко В.А. От многослойных персептронов к нейронным сетям глубокого доверия: парадигмы обучения и применение // XVII Всероссийская научно-техническая конференция с международным участием, 2015. – С. 47-84.
5. Гонсалес Р. Цифровая обработка изображений / Р. Гонсалес, Р. Вудс. – Москва: Техносфера, 2012. – 1104 с.
6. Каллан, Р. Нейронные сети: Краткий справочник / Р. Каллан. - М.: Вильямс И.Д., 2017. - 288 с.
7. Клейнберг Дж., Тардос Е. Алгоритмы: разработка и применение. Классика Computers Science / Пер. с англ. Е. Матвеева. – СПб.: Питер, 2016. – 800 с.
8. Красильников Н. Цифровая обработка 2D- и 3D-изображений / Н. Красильников: Отдельное издание. - БХВ-Петербург, 2011. - 608 с.
9. Редько, В.Г. Эволюция, нейронные сети, интеллект: Модели и концепции эволюционной кибернетики / В.Г. Редько. - М.: Ленанд, 2019. - 224 с.
10. Beyerer J. Automated Visual Inspection: Theory, Practice and Applications / J. Beyerer, P. Fernando, F. Christian. – Springer Berlin Heidelberg, 2016. – 798 p.

11. Calonder M. BRIEF: binary robust independent elementary features / M. Calonder, V. Lepetit, C. Strecha, P. Fua // European Conference on Computer Vision, 2010. – P. 778-792.

12. Chollet F. Deep Learning with Python – Manning Publications, 2017. – 384 p.

13. Geron Au. Hands-On Machine Learning with Scikit-Learn, Keras and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems Second Edition (Third Release) – O'Reilly Media, 2019. – 856 p.

14. Lubanovic B. Introducing Python: Modern Computing in Simple Packages Second Edition – O'Reilly Media, 2019. – 605 p.

15. Lutz M. Python Pocket Reference, 5th Edition – O'Reilly Media, 2014. – 264 p.

16. McKinney W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython, 2nd Edition – O'Reilly Media, 2017. – 544 p.

17. Muller A.C. Introduction to Machine Learning with Python: A Guide for Data Scientists / A.C. Muller, S. Guido – O'Reilly Media, 2016. – 392 p.

18. Singh P. Learn TensorFlow 2.0: Implement Machine Learning and Deep Learning Models with Python / P. Singh, A. Manure – Apress, 2019. – 177 p.

19. Sutskever I. On the importance of initialization and momentum in deep learning / I. Sutskever, J. Martens, G. Dahl, G. Hinton // Journal of Machine Learning Research. – 2013. – V. 28, No. 3. – P. 1139–1147.

20. VanderPlas J. Python Data Science Handbook: Essential Tools for Working with Data – O'Reilly Media, 2016. – 672 p.