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)

01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки / специальности)

Компьютерные технологии и математическое моделирование
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Применение методов машинного обучения для совмещения нескольких видеопотоков в общий поток»

Обучающийся Г. С. Поляков
(Инициалы Фамилия) (личная подпись)

Руководитель С. В. Митин
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Консультант Кандидат педагогических наук, Т. С. Якушева
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

АННОТАЦИЯ

Тема бакалаврской работы: «Применение методов машинного обучения для совмещения нескольких видеопотоков в общий поток».

В данной бакалаврской работе исследуются способы слияния нескольких видеопотоков в один общий видеопоток.

Цель работы - разработка приложения, для слияния нескольких видеопотоков с использованием методов машинного обучения.

За основу взята сверточная модель нейронной сети совместно со сторонними методами анализа и поиска связей между объектами двух кадров изображения для дальнейшего совмещения двух видеопотоков. В работе предложены варианты совмещения двух подходов для поиска лучшей концепции работы предложенного алгоритма. На языке Python разработана программа, которая демонстрирует возможности предложенного варианта реализации комбинации методов машинного обучения с аналитическими инструментами при решении задачи совмещения двух видеопотоков в общий поток. Работа приложения протестирована на различных изображениях.

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

Во введении описывается актуальность проводимого исследования, дается краткая характеристика проделанной работы, цель работы.

В первом разделе проводится анализ перспектив развития методов по совмещению видеопотоков.

Во втором разделе описываются различные методы комбинации изображений.

В третьем разделе дается описание разработанного приложения для совмещения видеопотоков.

В заключении представлены выводы по проделанной работе.

В работе использовано 12 рисунков, 9 формул, список литературы содержит 21 литературный источник. Объем бакалаврской работы составляет 40 страниц.

ABSTRACT

The subject of bachelor 's work is: "Appliance of machine learning methods to combine several video streams into a single stream".

The bachelor’s paper explores ways of combining several video streams into a single stream.

The aim of the work is to develop an application for junction of several video streams by using machine learning methods.

The convolutional neural network model is taken as a basis, together with third-party methods for analyzing and searching for links between objects of two image frames for further combining two video streams. The paper proposes options for combining the two approaches to find the best concept of the proposed algorithm. The program in Python that demonstrates the capabilities of the proposed implementation of a combination of machine learning methods with analytical tools in solving the problem of combining two video streams into a common stream. The application for various images has been tested.

The structure of the bachelor's work is represented by an introduction, three chapters, a conclusion, a list of references.

The introduction describes the relevance of the study, gives a brief description of the work done, the purpose of the work.

The first chapter analyzes the prospects for the development of methods for combining video streams.

The second chapter describes various methods for combining images.

The third chapter describes the developed application for combining video streams.

In conclusion, key conclusions for the graduation work done are presented.

The work used 12 figures, 9 formulas, the list of references contains 21 references. The volume of bachelor's work is 40 pages.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

С каждым годом появляется все больше сфер человеческой деятельности, в которых машинное обучение находит свое применение. Как пример, для управления крупногабаритным транспортом необходим особый контроль за слепыми зонами при вождении.

Это удобный инструмент, облегчающий процесс рутинной разработки тяжеловесных программ, задачей которых является решение сложных узконаправленных задач. Алгоритмы анализа изображений, например, ежедневно помогают не только самим программистам, облегчая их работу, но и обычным людям, предлагая им дешевый и доступный продукт. Многие компании вкладывают огромные суммы денег в разработку нового программного обеспечения, основанного на моделях машинного обучения, и эти инвестиции приносят прибыль этим субъектам. Именно поэтому область разработки с элементами систем искусственного интеллекта остается актуальной.

В данной бакалаврской работе исследуются способы совмещения нескольких видеопотоков в единый видеопоток на основе методов машинного обучения, а объектом исследования являются алгоритмы для выполнения операции склейки кадров и синхронизации видеопотоков.

Цель работы – разработка программы, которая позволяла бы совмещать несколько видеопотоков в один.

На языке программирования Python разработан программный модуль, который демонстрирует возможности методологий машинного обучения при решении задачи совмещения двух подвижных изображений. Работа программы была протестирована на различных видеорядах и изображениях.

1 Анализ проблемы совмещения видеопотоков

Видеопотоком являются беспрерывно воспроизводящиеся кадры определенного формата, закодированные в битовый поток. За единицу времени может воспроизвестись некоторое количество кадров изображения, у каждого из которых одинаковые параметры высоты и ширины картинки, из которых складывается размер единого кадра. Пропускная способность – соотношение предельного количества проходящих единиц информации в единицу времени через канал - различных видеосистем разнится от варианта к варианту исполнения системы. При анализе изображения огромное внимание к себе уделяет его размер. Чем больше деталей на изображении, больше его разрешение, тем больше и сам размер файла.

Обработка видео в последние десятилетия стало очень востребованной дисциплиной среди общей массы редактирования информации. Помимо наложения эффектов на видеоматериалы, активно применяются различного рода склейки кадров, будь то межкадровый и внутрикадровый монтаж, либо же частные методы решения изменения кадрового пространства видеопотока. К последним относится добавление новой информации к видеопотоку путем слияния с другим видеопотоком, при том таким образом, что итоговое изображение представляет собой компиляцию из двух разных кадров внутри одного кадра. Свое применение данная технология находит во многих сферах: начиная от потребительского сегмента, с системами помощи водителю, которые в свою очередь опираются на обзор дорожно-транспортной обстановки с помощью встроенных камер, изображение с которых склеивается в общий поток, и заканчивая промышленностью, где необходим контроль за производственным объектом путем видеосъемки широкого угла обзора. И если обычная видеосъемка часто справляется с поставленной задачей, то если дело касается подобного рода специфичных вопросов, то необходимы другие нестандартные решения.

Благодаря общему техническому прогрессу и непрекращающемуся консюмеризму по всему миру, технология склеивания изображений

становится все более актуальной. В решении поставленной задачи эффективными средствами являются методы, основанные на глубоком машинном обучении, которые с конца восьмидесятых годов прошлого века неумолимо увеличивают показатели востребованности от года к году в самых разных сфера жизнедеятельности человека.

Использование глубоких нейронных сетей при решении практических задач связано со следующими преимуществами:

— Программные продукты на основе нейронных сетей требуют от разработчика и исследователя гораздо меньше действий, чем нежели консервативные методы анализа и проектирования программ. Глубокие нейронные сети позволяют автоматизировать процесс подбора необходимых параметров для получения искомого или близкого к искомому результата.

— Архитектура алгоритмов нейронных сетей позволяют применять их для самых разных задач, поскольку они не привязаны к методологиям функционального программирования с жестким детерминированным отношения алгоритма к получаемому результату. Также, помимо всего прочего, работа нейронных сетей основана на нелинейном преобразовании данных. Популярность нейронных сетей также растет вследствие того фактора, что большая часть реальных задач имеют неудовлетворительную линейную аппроксимацию.

На основе вышесказанного можно сделать вывод, что с возрастающим использованием систем компьютерного зрения в различных сферах, возрастает так же потребность в нестандартных пользовательских решениях, одним из которых будет являться совмещение видеопотоков. Также функциональные возможности глубоких нейронных сетей в теории позволяют распознавать паттерны изображений, для поиска общей разницы между двумя кадрами разных видеопотоков. В рамках выполнения текущей бакалаврской работы планируется создать программное решение на основе глубокой нейронной сети для выполнения поиска общей разницы двух

кадров разных видеопотоков для дальнейшего позиционирования видеопотоков относительно друг друга.

Целью является разработка программного модуля, которая позволяла бы конечному пользователю совместить два видеопотока в один при помощи глубоких нейронных сетей.

Для реализации поставленной цели необходимо решить следующие задачи:

- Провести анализ проблем совмещения двух кадров из разных видеопотоков;
- Разработать метод применения нейронных сетей для поиска общих паттернов двух кадров разных видеопотоков;
- Спроектировать и разработать программный модуль для совмещения нескольких видеопотоков в общий поток.

Выводы по разделу 1

С непрерывно возрастающим консюмеризмом в различных сферах растет также потребность в решении специфических задач связанных с видеоинформацией, в число которых входит совмещение нескольких видеопотоков в один поток. При этом функциональные возможности методов машинного обучения в теории позволяют облегчить задачу позиционирования кадров изображения относительно друг друга. На основании вышеизложенных фактов была сформулирована цель работы и задачи, которые необходимо решить для ее достижения.

2 Применение методов машинного обучения для поиска общих паттернов двух кадров разных видеопотоков

2.1 Методы синхронизации видеопотоков

Для начала анализа материала и реализации алгоритмов поиска общих паттернов необходимо сопоставить два видеопотока в соответствии с записанным материалом – синхронизировать потоки.

При записи видео с двух и более камер высока вероятность разницы соответствующих кадров из-за отставания при начале записи на той или иной камере. Решением проблемы является программный анализ записанного материала.

Загружаемые пользователем видеофайлы имеют наборы описательных данных о файле, основными параметрами которых являются длительность записи, дата записи, тип файла, расположение, имя автора, информация об авторских правах и так далее. Нас же интересует в первую очередь дата записи видеофайла.

Дата записи видеофайла включает в себя набор числовых значений, разделенных на подпункты - непосредственно дата и время записи видеофайла.

Имея указанные данные, вычисляется разница между началом записи видеофайлов и по данной разнице из более раннего потока убирается лишняя часть, которая была записана за некоторое время до старта записи второго видеофайла.

Подобным образом происходит синхронизация видеоматериалов при наличии метаданных в виде даты и времени записи. Тем не менее, существует ряд случаев, когда в аппаратной части отсутствует функция фиксирования времени и даты, либо же данная функция работает некорректно вследствие человеческого фактора или влияния внешней среды. Для данных случаев описанный выше метод не будет работать.

В иных случаях необходимо прибегнуть к синхронизации при помощи метода нумератора. При начале записи с двух камер необходимо воспроизвести громкий короткий звук, который будет присутствовать в обоих видеозаписях. Это оставит звуковую сигнатуру с высокой амплитудой сигнала на обеих записях и позволит по ним синхронизировать кадры видеопотоков.

От подготовленных видеофайлов выделяется звуковая дорожка и разбивается на N-количество секторов, после чего на данные звуковые каналы применяется преобразование Фурье. Данное преобразование разложить функцию (звуковую дорожку), зависящую от времени, на другую функцию, зависящую от амплитудных значений – амплитудный спектр [5]. Преобразование Фурье функции f вещественной переменной задается следующей формулой [10]:

$$\hat{f}(\omega) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\infty} f(x) e^{-ix\omega} dx \quad (1)$$

где f(x) – входной сигнал,

$\hat{f}(\omega)$ – изображение по Фурье,

ω – частота.

По полученным значениям на отрезках выделяются частоты с наивысшей амплитудой.

По полученной карте значений наибольших частот происходит поиск подходящих значений соответствующих параметров после аналогичных преобразований на другом аудиофайле из иного видеоматериала.

2.2 Методы наложения изображений

Рассмотрим наиболее распространенные методы поиска общих паттернов изображений для их последующей склейки.

Основным методом, без участия в алгоритме машинного обучения, являются поиск общей разницы пикселей при пересечении двух изображений. Для этого пользователи зачастую используют программное

обеспечение для редактирования фотографий – Adobe Photoshop, Canva, GIMP и так далее.

Пользователь в новом проекте внутри программного продукта импортирует два изображения и пересекает их поверх искомой области общей суммы. Затем им применяется принцип наложения изображений с последующим подсчетом разницы пикселей данных изображений. При идеальном совмещении двух кадров, при совершенной идентичности их пересекаемых областей, разница пикселей сводит пересечение этих изображений к черному цвету. Это означает, что разница между информацией каждого пикселя первого кадра и соответствующего ему пикселя второго кадра равна нулю – в разнице при вычитании остается ноль, который и преобразует итоговое значение пикселя в черный цвет, отсутствие цветовой информации.

Данный метод имеет следующие недостатки:

- Необходимо ручное позиционирование кадров относительно друг друга;
- При отсутствии явной абсолютной разницы пользователь не сможет найти нужные параметры постановки кадров;
- Вся процедура многократно усложняется, если кадры необходимо позиционировать еще и в наклоне;
- Средства программного обеспечения не позволяют применить подобные операции к видеофайлам.

Помимо всего прочего, если пользователю удается достигнуть относительного минимума по разности информации пикселей двух кадров, то в итоге изображение, если разница не равняется нулю, будет иметь артефакты объектов, когда их черты искажаются и перестают быть похожими на реальные объекты. При решении данной проблемы, пользователю необходимо тратить большое количество времени и сил на устранение мелких проблем и недочетов, что все равно не гарантирует получение желаемого результата.

Помимо проблемы позиционирования изображения стоит также выделить сложности при обработке итоговой композиции. После выравнивания двух кадров, их общая область наложения все равно может незначительно отличаться друг от друга. Это обусловлено факторами в реальной жизни – степенью освещения, попаданием тени на матрицу одной камеры, автоматическими настройками камер (выбор экспозиции кадра, фокусного расстояния, баланса белого цвета изображения). При наличии разницы в параметрах настройки камер, итоговое изображение будет значительно отличаться, то не позволит без последующей обработки конкретного кадра идеально склеить изображения.

Решением данной проблемы является выставление четких требований при съемке видеоматериалов – настройки камер должны быть фиксированными и идентичными и располагаться они должны в непосредственной близости друг к другу.

Но если подобного рода проблемы не избежать, то необходимо принимать особые методы обработки итоговых изображений с двух камер. Одним из таких методов является сумма информации пересечения кадров по принципу темнейшего или светлейшего пикселя.

Берется информация о каждом пересекаемом пикселе и о каждом пересекающем пикселе в зоне пересечения изображений. Производится анализ данных между двумя пикселями – чья информация окажется более «темной» по отношению к другому, тот выступает в роли «внешнего» пикселя изображения. Но если пересекающий пиксель окажется светлее пересекаемого, то «внешним» станет пересекаемый. Подобным способом выстраивается правдоподобное изображение с минимальными потерями информации – картинка ровная, без дисторсий и сильных артефактов, аномалии почти отсутствуют. Подобное наложение изображений называется затемнением.

Тот же принцип работает можно сделать обратным и получить эффект более светлого пикселя – осветление.

Данные виды наложения являются неадаптивными алгоритмами, что описывает их работу таким образом, что к каждому пикселю пересечения применяются одинаковые методы. Адаптивные алгоритмы обработки изображений являются узкоспециализированными и особой популярности не имеют. Их наличие можно обнаружить лишь в лицензированных программах.

2.3 Методы поиска общих паттернов между двумя изображениями

Искусственные нейронные сети принадлежат сфере машинного обучения [2].

Машинное обучение – это раздел науки об искусственном интеллекте, который изучает алгоритмы способные обучаться на заданном наборе данных [7].

Алгоритмы машинного обучения принято делить на 2 категории:

- алгоритмы обучения без учителя – принадлежат классу задач обработки данных, в которых известны только значения атрибутов исходного множества объектов. Алгоритмы из данной категории, в ходе своей работы, сравнивают заданные объекты с целью поиска в них закономерностей и зависимостей;
- алгоритмы обучения с учителем – для таких алгоритмов подразумевается наличие пар «входные данные – правильный ответ». Алгоритм восстанавливает связь между входными данными и ожидаемыми результатами. На основе изученной информации алгоритм обучается создавать результат максимально схожий с правильным результатом тестовой выборки.

Искусственные нейронные сети представляют из себя вычислительные архитектуры, обрабатывающие поступающую информацию в ходе многочисленных вычислений функций матричных умножений и функций ошибок, для достижения желаемого результата. Процесс работы с

информацией во многом схож с человеческим мозгом, который также способен обучаться и анализировать большие и сложные наборы данных, которые с помощью более линейных алгоритмов обработать крайне нелегко [2], [3].

Как уже упоминалось ранее, особенностью нейронных сетей можно назвать их способность к обучению. Данный процесс становится возможным благодаря длинному итерационному процессу подбора параметра весов функциями подсчета ошибок, модулируя работу нейронной сети для достижения ожидаемого результата.

Обучение происходит по заранее подготовленным данным, определяющим дальнейшее поведение и назначение конкретной нейронной сети. Чем более полным и разнообразным будет набор обучающей выборки, тем более гибкой окажется вариативность ответов нейронной сети на запрос пользователя. Также большие объемы данных с общим признаком, то есть расширение обучающего множества, позволяют бороться с проблемой переобучения.

Уже обученная и настроенная нейронная сеть должна выдавать ожидаемый от нее результат при вводимых данных пользователем, при том, результат должен иметь общие черты с обучающей выборкой, но не являться ее частью.

Вычислительная единица, которая получает информацию, производит над ней вычисления и передает ее дальше называется нейроном [1][11]. Они делятся на три основных типа: входной, скрытый и выходной.

По сути своей это нелинейная функция, задача которой заключается в обработке входных сигналов и возврат значения на выход к другому нейрону или слою, следующему за ним. Данная функция называется функцией активации.

Связь между двумя и более нейронами называется синапсом. При том, синапсом является связь как со следующими, так и с предыдущими нейронами внутри нейронной сети. У синапсов есть 1 параметр — вес.

Благодаря ему, входная информация изменяется, когда передается от одного нейрона к другому.

В том случае, когда нейросеть состоит из большого числа нейронов, вводится понятие слоев, представляющих из себя комбинацию наборов множества нейронов. Слои чередуются друг за другом в зависимости от их функциональности и задач. Первый слой называется входным слоем.

Входной слой получает информацию и распределяет ее для дальнейшей обработки внутренними или скрытыми слоями нейронной сети. Входные нейроны являются объединёнными с основным слоем с помощью синапсов с разными весами, обеспечивающими качество связей [1][2][11].

Последний слой нейронной сети называют выходным слоем. Данный слой предоставляет обработанные данные в качестве выхода результата работы общего алгоритма сети. Этот результат напрямую зависит от функций весов, активации и ошибок внутри нейронной сети.

Внутренние или скрытые слои нейронной сети находятся между входным и выходным слоями. В них происходит вся обработка информации, поступаемой в сеть. Также функции ошибок корректируют их работу для получения корректного желаемого результата работы сети [2].

Слой, выходные нейроны которого связаны со всеми входными нейронами, называется полносвязным. Выходом полносвязного слоя является не одно число, а вектор. Выходной слой всегда является полносвязным поскольку все нейроны этого слоя имеют полное соединение со всеми функциями активации предыдущего слоя.

Обучение внутри слоев происходит за счет подбора правильных значений весов функцией потерь или ошибок. Она показывает разницу между реальными данными, вводимыми пользователем, и получаемыми в ходе выполнения программного модуля.

Нейронные сети различаются по моделям, представлением структуры. В том случае, если у нейросети три и более слоя, то такую сеть называют глубокой нейронной сетью. Большее количество слоев ведет к более

сложным и точным результатам работы алгоритма. Также на это влияет и итоговое количество нейронов и синапсов внутри сети.

При анализе изображений и поиске общих паттернов среди них чаще всего используются сверточные нейронные сети [1][2]. Вследствие стремительного прогресса в производстве чипов и наращивания мощности конечных изделий в сфере микроэлектроники, в последнее время данные нейросети пользуются особым успехом так же, как и задачи компьютерного зрения, классификации изображений. Не малую роль сыграло также наличие большого числа обучающих выборок для тренировки нейронных сетей.

Модель сверточных нейронных сетей прежде всего отличает ее архитектура. Идея этих сетей заключается в чередовании сверточных слоев и субдискретизирующих слоев, то есть слоев подвыборки. Структура данной сети является однонаправленной, без обратных связей, и многослойной [7].

Название данная архитектура сети получила вследствие наличия в ней операции свертки, когда каждый фрагмент изображения умножается на матрицу свертки поэлементно, а результат суммируется и записывается в аналогичную позицию выходного изображения.

Также, помимо этого, скрытые слои нейронной сети состоят из множества карт признаков, в которых все нейроны имеют одинаковые веса. За операцию усреднения карт отвечает вычислительный слой, который идет вслед за слоем свертки. Данный слой служит для нормирования значений по признакам определенной функции конкретного слоя [7][11].

Функцией слоев субдискретизации является многократное уменьшение размера карт признаков для ускорения дальнейших вычислений. Общие характеристики значения признака в задаче компьютерного зрения не так важны, как наличие самого признака, что делает возможным операцию по сокращению качества обрабатываемой информации в угоду скорости выполнения вычислений [1][3][11].

Как уже описывалось ранее, суть сверточных нейросетей заключается в чередовании слоев свертки и подвыборки, при чем, стоит отметить возможность их произвольного чередования. Порой это необходимо для распознавания более сложных образов, иерархических признаков и позволяет формировать представление об абстрактных деталях входных данных. Достигается данный эффект при чередовании карт признаков друг за другом [6][8].

В задаче поиска общих паттернов изображений возможно использование следующих технологий и методологий [9]:

— распознавание примитивов на изображении;
— Евклидовых метрик для определения расстояния между двумя распознанными объектами;
— извлечение признаков с помощью фильтров;
— сверточные слои нейронной сети.

Первым этапом будет формирование признаков при помощи фильтров. Первый слой нейронной сети принимает все пиксели в изображении. После того, как все данные введены в сеть, к изображению применяются различные фильтры, которые формируют понимание различных частей изображения.

Вторым этапом будет выделение патчей и их представление. На этом этапе проводится получение отдельных участков из изображения и представление их в виде многомерных векторов содержащих карты признаков размером равным размерности вектора по формуле (1):

$$F_1(y) = \max(0, W_1 * y + B_1) \quad (2)$$

где W_1 – фильтр,

$B_1 \in R^{n1}$ – вектор сдвига,

символ «$*$» - операция свертки.

При этом:

$$W_1 = c * f_1 * f_1 \quad (3)$$

где c – число каналов на изображении,

f_1 – размер фильтра,

n_1 – число операций свертки.

Взятие поэлементного максимума осуществляется с использованием функции активации сверточной нейронной сети [19][20]. Веса B_1 представляют собой n_1-мерный вектор, каждый элемент которого сопоставлен с элементом фильтра W_1.

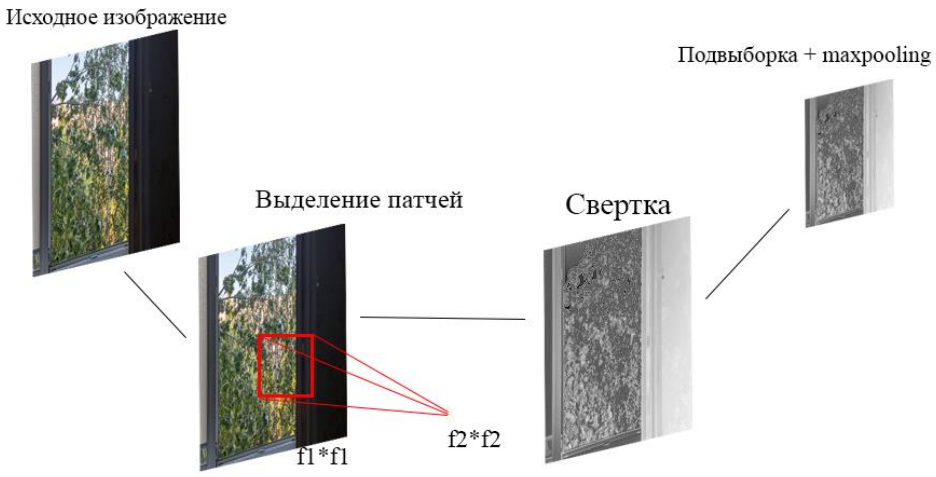

Рисунок 1 – Сверточная нейронная сеть

Третьим этапом будет подвыборка – операция уменьшения размерности, которая выполняет уменьшение размерности сформированных карт признаков [8]. В данной архитектуре сети считается, что информация о факте наличия искомого признака важнее точного знания его координат, поэтому из нескольких соседних нейронов карты признаков выбирается максимальный и принимается за один нейрон уплотнённой карты признаков меньшей размерности. За счёт данной операции, помимо ускорения дальнейших вычислений, сеть становится более инвариантной к масштабу

входного изображения. Чаще всего – выбор максимума или поиск среднего по нескольким соседним нейронам [11].

Чередование слоев свертки и подвыборки формирует сверточную глубокую нейронную сеть. Все это необходимо для упрощения исходного изображения до примитивов, анализ которых происходит внутри модели выборки. Распознавание более сложных и больших объектов требует большего времени тренировки, поэтому необходимо упростить исходное изображение через его фильтрацию [2].

Если оба изображения имеют общие черты, признаки, тогда общие операции фильтрации на них проявят одинаковые объекты, распознав которые программа будет иметь представление о том, где они находятся в каждом кадре для их дальнейшего позиционирования по методу Евклидовых метрик.

Структура глубокой нейронной сети представляет из себя чередование слоев выделения отдельный патчей изображения, слоев свертки и слоев пуллинга с подвыборкой. Данная модель позволяет нам распознавать образы на отдельных участках изображения для их дальнейшего анализа данной нейронной сетью и сравнения с имеющимися представлениями на соседнем изображении. В результате нейронная сеть определяет схожесть сущностей двух кадров по отдельным признакам, вскрывающимся на слоях свертки с фильтрацией.

Во всей структуре также присутствуют функции активации (ReLU). Они представляют из себя определение выходных значений нейрона в зависимости от результатов взвешенной суммы входов и порогового значения. Благодаря данной функции активации нужные нам значения включаются заметно раньше чем нежели при использовании линейной функции или сигмоиды [21]. Положительные выходные параметры преобладают над отрицательными из-за отсутствия сигнала функции активации к отрицательным значениям:

$$R(z) = \max(0, z) \quad (4)$$

где max – функция выбора по максимуму,

z – параметр функции.

При положительных значениях z функция возвращает значение z. При отрицательных возвращает ноль.

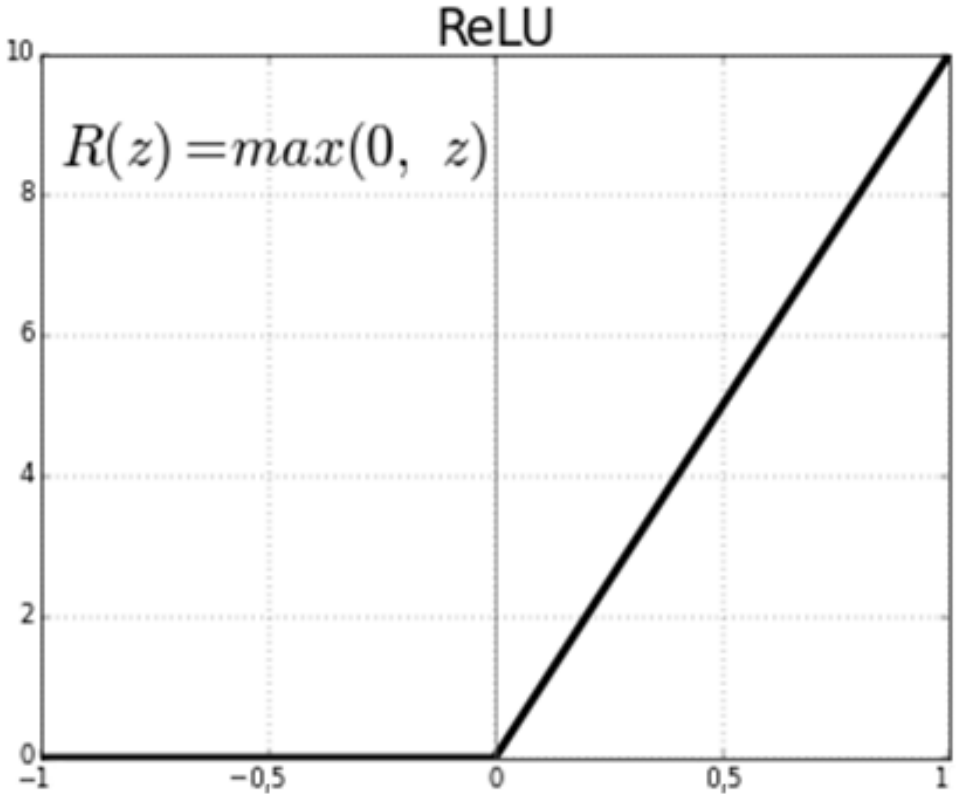

Рисунок 2 – График функции ReLU.

Обучение модели происходит за счет заранее подготовленного датасета состоящего из тысяч изображений примитивов после этапов фильтрации.

Общая задача совмещения двух видеопотоков при помощи методов машинного обучения низводится до поиска общих паттернов у двух кадров из целевых видеопотоков для их дальнейшего совмещения с наложением одного видеопотока на другой. То есть, вся обработка делится на два больших шага:

— Анализ изображений, поиск общих паттернов. Нахождение желаемого вектора смещения кадров.

— Применение склейки двух видеопотоков посредством смещения одного видеопотока относительно другого с последующей обработкой секции пересечения кадров функциями наложения изображений.

Входные данные – первые кадры из видеопотоков или отдельные изображения, выбранные пользователем. У необходимых кадров выделяется необходимая область поиска смежных значений кадров [18]. Для этого по умолчанию берется половина вставленного кадра по его ширине – они будут являться входными значениями для нейронной сети. У каждого изображения происходит многократная фильтрация данных для выделения паттернов, по которым будет происходить поиск общих черт двух изображений. На основании полученных данных, формируется карта признаков [3][4][17]:

$$x_j^l = f(\sum_i x_i^{l-1} * k_j^l + b_j^l), \quad (5)$$

где x – карта признаков j, выход слоя l;

f() – функция активации;

b – коэффициент сдвига слоя l карты признаков j;

k – ядро свертки карты j, слоя l;

* - операция свертки входа x.

Из библиотеки Keras также используется методы подсчета распределения вероятностей при помощи функции softmax – выводит в качестве результата вектор, представляющий список распределения вероятностей потенциальных результатов [14]. Преобразует входные данные (логиты) в вероятности, получая экспоненту e каждого значения, а затем подвергая нормализации каждое значение таким образом, чтобы сумма всех экспонент равнялась единице [11][12]:

$$S(y) = (e^{y(i)})/(\sum_{j=1}^{n} e^{y(i)}), \quad (6)$$

где y – входные значения;

S(y) – функция softmax;

e – экспонента;

i – конкретный сигнал нейрона.

Получается законченная нейронная сеть, задача которой сводится к классификации паттернов изображений и их дальнейшего анализа.

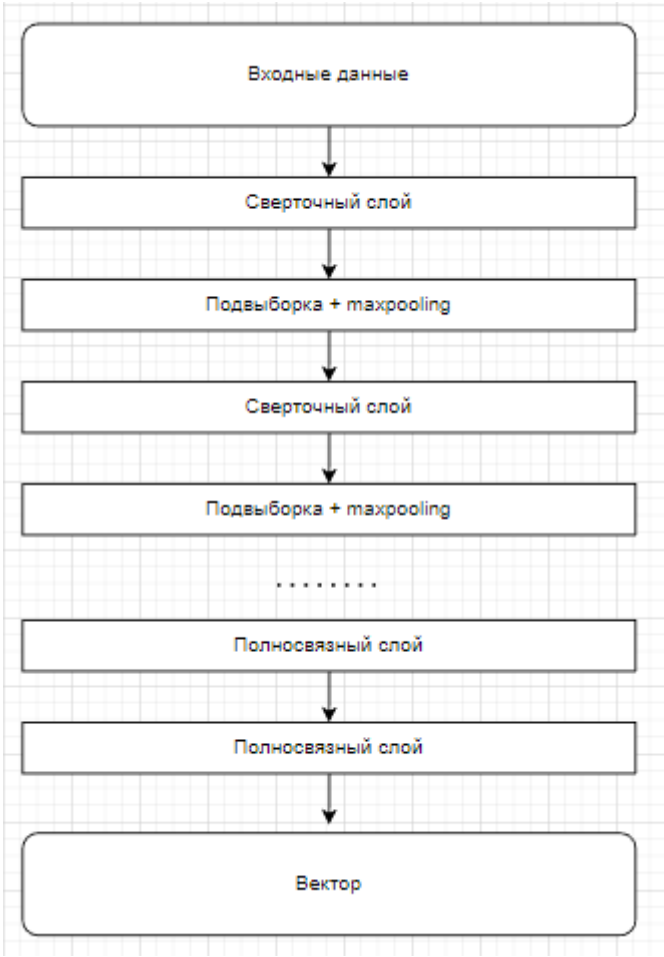

Рисунок 3 – схема сверточной нейронной сети.

Полученная матрица векторов является категориальным представлением признаков паттерна на конкретном участке обхода изображения. Поскольку работа нейросети представляет из себя обход двух изображений, то и векторов будет два. Обе матрицы векторов в дальнейшем подвергаются сравнению для нахождения схожих паттернов для дальнейшего склеивания изображения.

Сравнение матричных представлений сводится к итеративному прохождению по размерностям матриц, их наполнению, для отслеживания схожестей паттернов. Если информация с текущего вектора сопоставима со

сравнимым вектором, то данный паттерн можно назвать схожим и информация об их местоположении в матрицы будут являться ключами для расчета смещения кадров изображений [16].

2.4 Процесс совмещения кадров

В результате работы нейронной сети, на выход приходятся данные о местоположении схожих паттернов двух изображений. Расположение паттернов определяет конечное смещение кадров относительно друг друга – в качестве поиска итогового смещения используются Евклидовы метрики.

Метрика в евклидовом пространстве, свойства которого описываются евклидовой геометрией, определяет расстояние между двумя точками данного пространства:

$$d(p,q) = \sqrt{\sum_{k=1}^{n}(p_k - q_k)^2}, \quad (7)$$

где p – координата первой точки,

q – координата второй точки.

Расчет евклидовой метрики также необходим при анализе итогового количества потерянных данных при слиянии двух кадров – чем больше расстояние, тем больше потери.

Расчет происходит с учетом расположения кадров вплотную относительно друг друга, как если бы они были единым изображением. Эта позиция и является стартовой при подсчете итогового вектора смещения первого кадра относительно второго.

Вектор смещения является набором разницы координат объекта первого кадра относительно второго. Данные параметры используются для применения операции наложения кадров друг на друга.

Рисунок 4 – иллюстрация нахождения общих паттернов и их соотношение друг с другом.

Наложение одного изображения на другое всегда неминуемо влечет за собой потерю или искажение конечно информации. Если разница между кадрами велика, то и потеря или искажение информации будет выше. Потерю информации при пересечении кадров можно уменьшить при использовании специальных параметров наложения изображений.

Операция наложения кадров описывается расчетом суммы или разницы информации в каждом пикселе двух и более изображений. Конечная математическая операция зависит от выбранного параметра наложения кадров друг на друга. При наложении как с разницей, так и с суммой итоговое изображение будет явно вскрывать разницу между двумя кадрами.

Рисунок 5 – наложение кадров с последующим поиском разницы.

Задачей же текущей работы является снижение искажений и уменьшение разницы между накладываемыми объектами, потому лучшей операцией будет наложение с темнейшим пикселем - Darken.

Суть данного наложения заключается в выборе минимального значения показателей каждого пикселя из пары двух значений – первого кадра и второго кадра.

$$I = \min(f_1, f_2), \tag{8}$$

где I – значение информации выходного пикселя,

f_1 – пиксель накладываемый,

f_2 – пиксель, на который накладывается изображение.

В итоге можно получить изображение, которое без детального осмотра будет казаться оригинальным по качеству предоставляемой информации для обозревателя.

Рисунок 6 – наложение кадров методом Darken.

Дальнейшая обработка склейки сводится к формированию прямоугольного представления изображения. Обрезаются лишние края на горизонтальных частях. Производится анализ потерянных данных при работе алгоритма. Учитываются площади пересекаемых областей, обрезанные части изображения.

$$F = \frac{S_0 - s}{S} * 100\%, \quad (9)$$

где F – процент остаточных данных(больше – лучше),

S_0 – конечная площадь изображения склейки кадров,

s – половина площади пересекаемой области,

S – сумма площадей двух начальных кадров.

Данные операции применяются для первых кадров потокового видеофайла, в качестве калибровочных данных. В ходе процесса калибровки отбираются параметры для дальнейших манипуляций с видеопотоками: параметры смещения, обрезки. Функция Darken также применяется ко всем промежуточным кадрам.

Выводы по разделу 2

Рассмотрены классические методологии машинного обучения и построения нейронных сетей. Также были описаны стандартные методы и технологии наложения изображений, основанные как на аналитическом методе, так и на методах машинного обучения. Для решения поставленной задачи предложена структура глубокой нейронной сети. Описаны методы наложения кадров и оптимального расчета их разницы. Выбрана модель расчета разницы для решения поставленной задачи.

3 Программная реализация модуля для совмещения нескольких видеопотоков в общий поток

3.1 Подготовка данных

Обучение глубоких нейронных сетей лучше проводить на материалах с высоким содержанием объектной базы. Также данные должны иметь одинаковые характеристики размера, формата и качества.

Одним из способов получения данных низкого разрешения для дальнейшей обработки нейросетью является подготовка данных высокого разрешения с дальнейшим дескейлингом параметров изображений – сохранением общих признаков, но уменьшением исходного разрешения картинки. Также зачастую применяются методы наложения шума или размытия исходного изображения [14].

Есть свободные ресурсы с базами изображений, такие как CIFAR-10, MNIST, Labelme. Большинство изображений там в виде .gz или .zip архивов.

Лучше всего для обучения нейронных сетей подойдут изображения в высоком разрешении цветового формата RGB [6]. Данный формат является оптимальным решением в вопросе соотношения качества предоставляемой информации к единице объекта, хранимой в памяти компьютера. К тому же данный метод легче всего обрабатывать программными методами вследствие наличия обширной базы методологий.

3.2 Средства разработки программного обеспечения

Множество языков программирования используются разработчиками для решения задач машинного обучения и обработки изображений. Существует также множество различных библиотек, которые уже содержат в себе наиболее часто используемые методы и алгоритмы.

Языком для написания программного модуля был выбран Python версии 3.9. Это решение обладает несколькими преимуществами в сравнении с другими вариантами:

— поскольку данный язык является одним из самых популярных в мире в текущий момент, на его основе реализовано огромное количество алгоритмических модулей и пакетов языка, которые позволяют облегчить работу с написанием кода программы;
— также на Python велико содержание библиотек машинного обучения. Менеджеры пакетов еще сильнее облегчают разработку конечного продукта;
— синтаксис языка достаточно прост и позволяет лишний раз избежать проблем при отладке конечного программного модуля из-за наличия автоматической сборки «мусора» в памяти программы и подсчета ссылок;
— каждый файл Python с разрешением .py является исполняемым, что увеличивает модульность программы и позволяет работать с каждым файлом как с отдельным скриптом.

Python зачастую не является основным выбором многих специалистов вследствие того, что данный язык является интерпретируемым, а значит скорость работы реализованных алгоритмов на его основе будет ниже, чем у компилируемых языков программирования, код которых сразу транслируется на ассемблерный код.

Тем не менее, в нагруженных частях программы возможно использование предкомпилированных библиотек на сторонних языках программирования, в частности на Rust.

Для программной реализации были задействованы следующие пакеты:

— Библиотека OpenCV – различного рода преобразования над кадрами видеопотоков;
— Библиотека NumPy – различного рода математические операции для промежуточных значений;
— Библиотека PyQt – графический интерфейс итогового программного модуля;

— Библиотека Keras – пакет методов на платформе машинного обучения Tensorflow для реализации нейронных сетей.

NumPy – это матетическая библиотека со множеством надстроек на ее основе. Она включает в себя множество методов по работе с многомерными структурами данных, массивами. В данной библиотеке также реализовано множество таких полезных вещей, как функции линейной алгебры, преобразования Фурье, генерации случайных чисел и так далее.

Как упоминалось ранее в этом разделе, Python, будучи интерпретируемым языком программирования, имеет тенденцию обрабатывать сложные математические алгоритмы медленнее, чем компилируемые языки. Авторы NumPy постарались разрешить эту проблему, предложив модуль с исполняемым кодом функций на языках C и Fortran.

Библиотека NumPy настолько популярна, что зачастую интегрирована в другие библиотеки. Так, например, функции из библиотеки OpenCV могут обрабатывать массивы из библиотеки NumPy.

OpenCV – это библиотека с открытым исходным кодом, содержащая наборы специальных типов данных и методы для работы с изображениями. Написана и оптимизирована на C/C++, также имеет интерфейсы для Python, Java, Ruby, Lua и других языков.

Эта библиотека представляет из себя набор модулей. Каждый из этих модулей связан с определенной областью компьютерного зрения.

Алгоритмы, реализованные в этой библиотеке, были протестированы и оптимизированы, что означает, что они, как и библиотека NumPy, могут частично снизить потери производительности, связанные с использованием интерпретатора Python. Библиотека включает в себя более 1000 функций и алгоритмов, решающих различные поставленные задачи.

OpenCV является достаточно широко используемой библиотекой благодаря ее свободной лицензии BSD. Она применяется такими компаниями, как NVidia, WillowGarage, Intel, Google. Компании NVidia и WillowGarage частично спонсируют ее разработку.

Keras – библиотека машинного обучения для языка Python с открытым исходным кодом. Используется для решения самого широкого спектра задач, к которым могут быть применимы методы машинного обучения: компьютерное зрение, обработка естественного языка, синтез речи и так далее. Ее код размещен в открытом репозитории на GitHub, так что любой желающий может попробовать внести некоторые изменения в исходных код библиотеки. Данный пакет является надстройкой над платформой TensorFlow, открытой программной библиотекой, разрабатываемой компанией Google. Keras является представляет из себя высокоуровневый набор абстракций, который делает простым формирование нейронных сетей, независимо от используемой в качестве вычислительного бэкенда библиотеки научных вычислений [13].

Большая часть приложений и программ, которые создаются в текущий момент, имеют у себя графический пользовательский интерфейс.

Пользовательский интерфейс включает в себя окна, кнопки, поля ввода текста, скроллеры, списки, переключатели, флажки и т.д., которые пользователь видит на экране при открытии приложения. Через них пользователь взаимодействует с программой и управляет ею.

Существует множество библиотек графического интерфейса пользователя. Tkinter является далеко не самой популярной, хотя с ее помощью написано множество различных проектов. Установочный файл Python обычно уже включает пакет Tkinter как часть стандартной библиотеки вместе с другими модулями.

Tkinter хорошо подходит для создания небольших графических модулей, модальных окон и легкого пользовательского интерфейса, но библиотека PyQt больше подходит для больших проектов. PyQt это инструмент для взаимодействия с графическим фреймворком Qt, реализованный в виде расширения Python.

PyQt также включает в себя Qt Designer (Qt Creator) — дизайнер графического интерфейса пользователя. Программа pyuic генерирует Python

код из файлов, созданных в Qt Designer. Это делает PyQt очень полезным инструментом для быстрого прототипирования. Кроме того, можно добавлять новые графические элементы управления, написанные на Python, в Qt Designer.

3.3 Структура программы

Структура представлена на рисунке

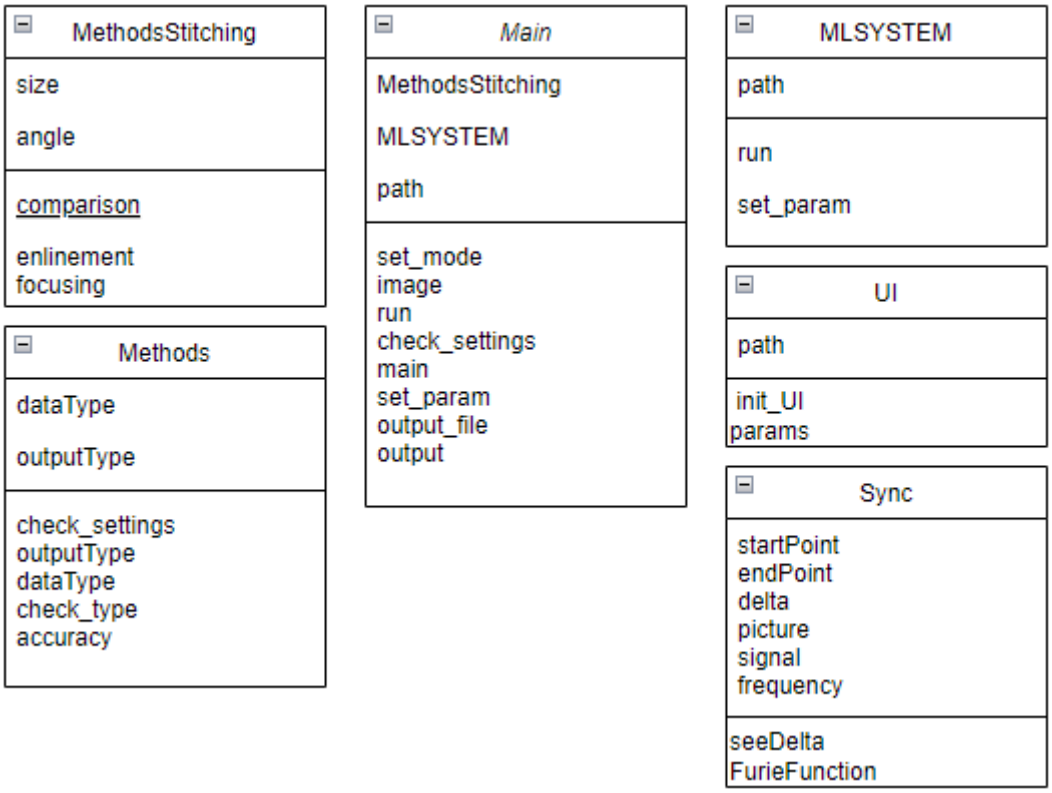

Рисунок 7 – Структура программного модуля

Приложение состоит из 6 программных блоков:

— Main – главный программный блок, инициализирует запуск программы. Наследование от UI.

— UI – программный блок, создаваемый пакетом PyQt. Содержит в себе все элементы графического интерфейса программы. Создает списки действий, интерфейс.

— MLSYSTEM – программный блок, работающий с преобразованными подготовленными изображениями.

— MethodsStitching – программный блок, работающий с выходными данными для совмещения изображений.

— Methods – программный блок, хранящий пользовательский ввод методологий и их функционал.

— Sync - программный блок, синхронизирующий два видеопотока по звуковой сигнатуре.

3.4 Графический интерфейс программы

При запуске программы появляется окно, представленное на рисунке

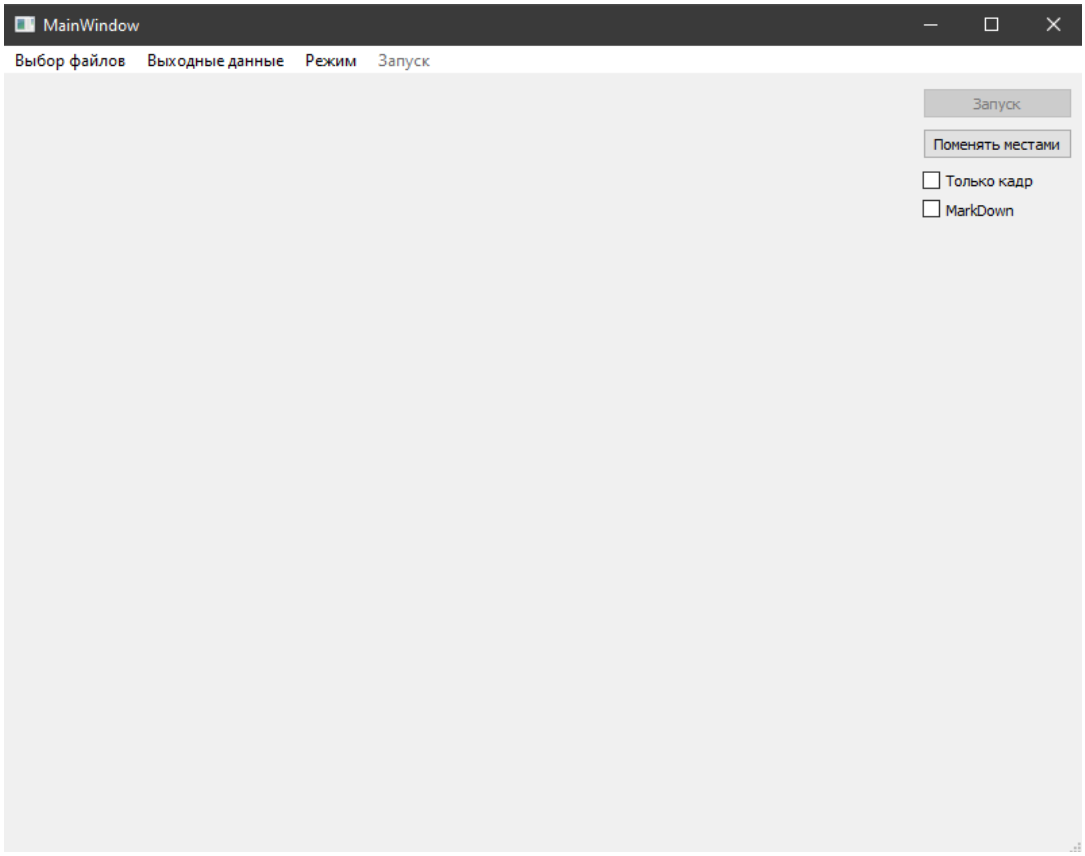

Рисунок 8 – Главное окно программного модуля

Главное окно представляет из себя рабочую область над которым расположены вкладки меню со следующими вариантами опций:

— Выбор файлов – открытие окна проводника Windows для выбора исходных файлов для работы программы;

— Выходные данные – открытие диалогового окна для выбора выходных данных в результате работы программного модуля;

— Режим – открытие диалогового окна для выбора режима сравнения изображений;

— Запуск – запуск работы программного модуля. При выборе данной опции без предварительной настройки предыдущих пунктов будут использоваться значения по умолчанию. Без выбора исходных файлов данный пункт будет неактивным для взаимодействия.

В результате работы программного модуля из исходных файлов будет получен набор данных, состоящий из следующих пунктов:

— Выходная видеозапись;

— Выходное изображение;

— Markdown-файл с параметрами смещения, оценкой степени потери данных изображений и временем работы программы.

3.5 Результаты тестирования программного модуля

Приведем несколько примеров работы приложения. Примеры показаны на следующих рисунках:

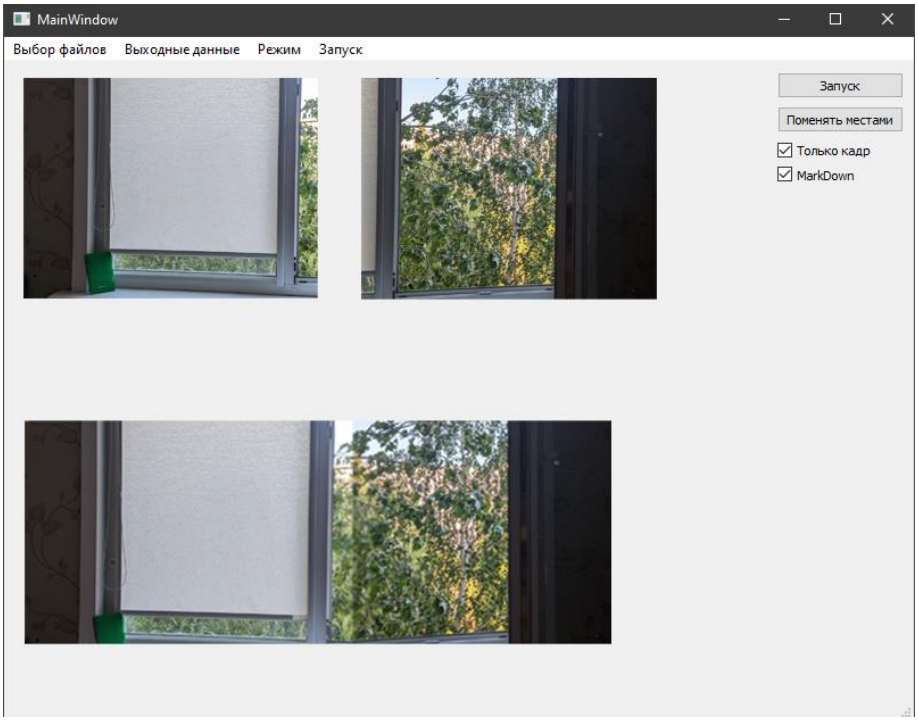

Рисунок 9 – Пример работы программного модуля с двумя отдельными изображениями

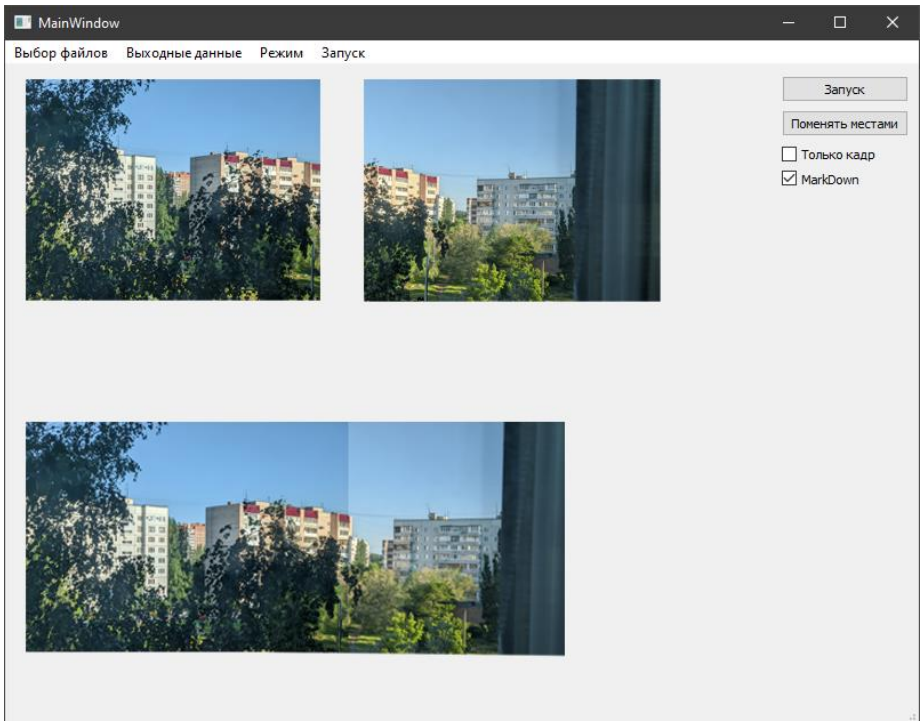

Рисунок 10 – Пример работы программного модуля с двумя отдельными видеофайлами

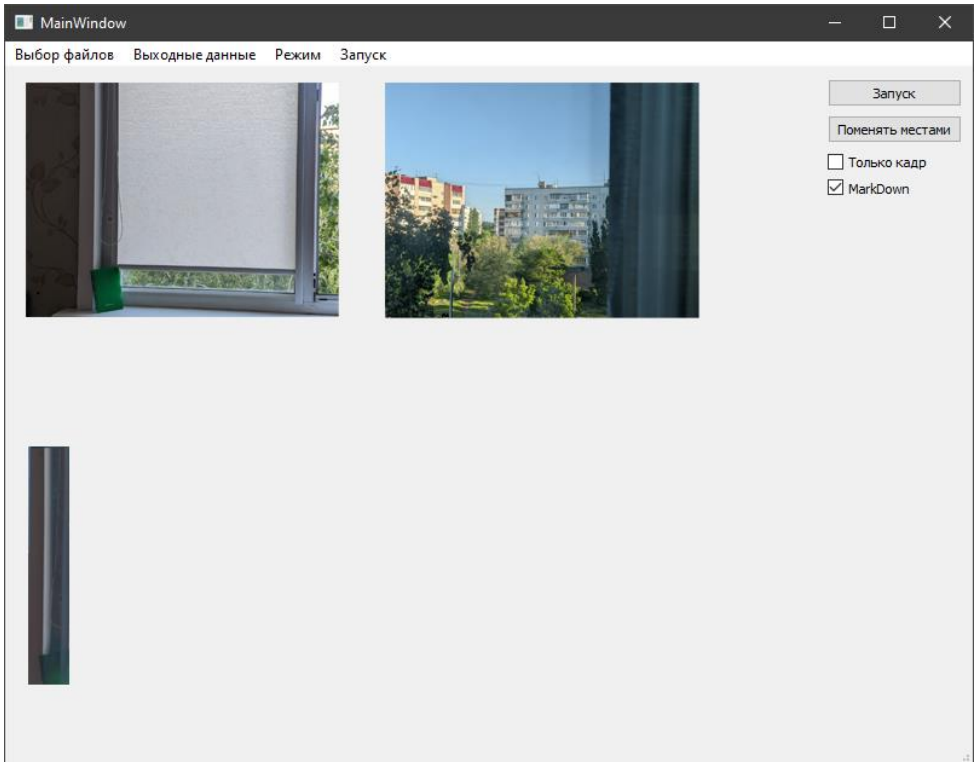

Рисунок 11 – Пример работы программного модуля при некорректном вводе данных. Левое изображение кардинально отличается от правого, вывод некорректен

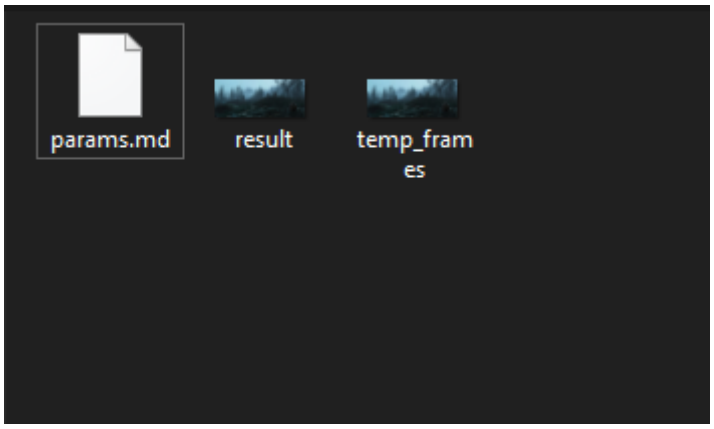

Рисунок 12 – Пример вывода программы после окончания работы

Как видно на представленных выше рисунках, разработанное приложение успешно справляется с поставленной задачей.

Выводы по разделу 3

На языке программирования Python разработано приложение с графическим интерфейсом позволяющее совмещать несколько видеопотоков с использованием работы глубоких нейронных сетей. Для моделирования работы слоев нейронной сети используется пакет Keras. Работа приложения протестирована на различных изображениях и видеофайлах.

Заключение

В ходе выполнения выпускной квалификационной работы были получены следующие результаты:

- Был проведен анализ методов машинного обучения и выбрана основная модель для решения поставленной задачи.
- В ходе выполнения работы был изучен и представлен материал об основных методах наложения изображений друг на друга.
- В ходе анализа теоретических материалов выяснилось, что при решении задачи поиска общих паттернов на изображениях наиболее эффективными являются сверточные нейронные сети.
- Синхронизация двух видеопотоков достигалась за счет анализа звуковой дорожки двух видеофайлов и последующим поиском общих паттернов в выделенных сигнатурах.
- Основным методом наложения кадров видеопотоков друг на друга выбран поиск разницы по темнейшему или светлейшему пикселю, поскольку данный метод позволяет выполнить наложение с меньшими визуальными искажениями.
- Был разработан программный модуль на языке программирования Python, позволяющий производить склейку нескольких видеопотоков в один поток с использованием методов машинного обучения, а в частности с применением сверточной нейронной сети.
- Программный модуль был протестирован на базе нескольких входных видеопотоках и изображений. Результаты тестирования считаются положительными, что доказывает эффективность применения нейронных сетей в решении проблемы совмещения видеопотоков.

Список используемой литературы

1. Власов А.В. Машинное обучение применительно к задаче классификации семян зерновых культур в видеопотоке / А.В. Власов, А.С. Федеев // Молодежь и современные информационные технологии – сборник трудов XIV Международной научно-практической конференции студентов, аспирантов и молодых учёных, 07–11 ноября 2016. – Национальный исследовательский Томский политехнический университет (Томск), 2016. – с. 133-135. – Текст : непосредственный.
2. Негру А. С. Увеличение разрешения изображений с использованием нейронных сетей // Издательство: Тольяттинский государственный университет, Институт математики, физики и информационных технологий, Кафедра Прикладная математика и информатика, 2020. – с. 40. – Текст: непосредственный.
3. Дорогов А.Ю. Нейронные сети глубокого обучения с управляемой коммутацией нейронных плоскостей / А.Ю. Дорогов // Дистанционные образовательные технологии: Материалы IV Всероссийской научно-практической конференции (с международным участием), 2019. – Издательство: Общество с ограниченной ответственностью «Издательство Типография «Ариал» (Симферополь), 2019. – с. 284-296. – Текст : непосредственный.
4. Хачумов М.В. О выборе метрики для решения задач классификации и кластеризации. – Материалы Первой всероссийской научной конференции с международным участием (SASM-2011) «Системный анализ и семиотическое моделирование» (Казань, 24-28 февраля 2011г.) – Казань: Издательство «Фэн» Академии наук РТ, 2011, с.255-260 – Текст : непосредственный.
5. Жуков, Д.А. Формирование контрольных выборок при технической диагностике объекта с применением машинного обучения / Д.А. Жуков, А.С. Хорева, Ю.Е. Кувайскова, В.Н. Клячкин // Математические методы и модели: теория, приложения и роль в

образовании – международная научно-техническая конференция : сборник научных трудов, 28–30 апреля 2016 года. – Ульяновский государственный технический университет (Ульяновск), 2016. – с. 44-48. – Текст : непосредственный.

6. Оппенгейм А.В. Цифровая обработка сигналов / Оппенгейм А.В., Шафер Р.В., Шац С.Я. // 1979. – Издательство: Связь, Москва 2018. – с. 416. – Текст : непосредственный.
7. Урубкин, М.Ю. Нейронные сети Кохонена и нечеткие нейронные сети в интеллектуальном анализе данных / М.Ю. Урубкин, А.В. Авакьянц // Совершенствование методологии познания в целях развития науки: сборник статей по итогам Международной научно-практической конференции: в 2 частях. 2017. – Издательство: Общество с ограниченной ответственностью "Агентство международных исследований" (Уфа), 2017. – с. 36-39. – Текст : непосредственный.
8. Якимчук, А.А. Глубокое обучение как эффективный метод машинного обучения / А.А. Якимчук // Научное сообщество студентов XXI столетия. Технические науки – сборник статей по материалам XCII студенческой международной научно-практической конференции. 2020. – ООО “Сибирская академическая книга” (Новосибирск), 2020. – с. 40-43. – Текст : непосредственный.
9. Крылов, А.С. Регуляризирующие методы интерполяции изображений / А.С.Крылов, А.В.Насонов. – М : АРГАМАК-МЕДИА, 2014. – 100 с. – Текст : непосредственный.
10. Насонов А.В. Развитие методов повышения качества изображений лиц в видеопотоке / А.В. Насонов, А.С. Крылов, О.С. Ушмаев // Информатика и ее применения, 2009. – Т. 3, Вып. 1. – С.19-28. – Текст : непосредственный.
11. Кандидов В.П. Дискретное преобразование Фурье / В.П. Кандидов, С.С. Чесноков, С.А. Шленов // Учебно-методическое пособие, 2019. – Москва: физический факультет МГУ, С.88. – Текст: непосредственный.

12. Rashid T. Make Your Own Neural Network // Publisher - CreateSpace Independent Publishing Platform; 1st edition, 2016, 222 pages – Text: direct.
13. Sebastian R. Python Machine Learning/ Sebastian Raschka, Vahid Mirjalili // 2019. Publisher - Packt Publishing, 2019. – Text : direct.
14. Geron A. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. // Publisher - O'Reilly Media; 2nd edition, 2019. 856 pages. Text: direct.
15. Gerardus B. Processing Image Recognition Standard Requirements // Publisher - 5STARCooks, 2022, Proceedings, pp. 309. – Текст : непосредственный.
16. Szeliski R. Computer Vision: Algorithms and Applications // Publisher : Springer; 2011, pp. 832. – Текст : непосредственный.
17. Sassatelli L. User-Adaptive Editing for 360 degree Video Streaming with Deep Reinforcement Learning / Lucile Sassatelli, Marco Winckler, Thomas Fisichella, Ramon Aparicio-Pardo // 27th ACM International Conference on Multimedia, Oct 2019, Nice, France. pp.2208-2210. – Текст : непосредственный.
18. Wei Di. Deep Learning Essentials / Wei Di, Anurag Bhardwaj, Jianing Wei // 2018. – pp. 482-491. – Текст : непосредственный.
19. Sohee Park. Advancing User Quality of Experience in 360-Degree Video Streaming With Machine Learning / Sohee Park, Arani Bhattacharya, Zhibo Yang // Published in: IEEE Transactions on Network and Service Management, March 2021. – pp. 1000 - 1015. – Текст : непосредственный.
20. Ajay Divakaran. Video mining: pattern discovery versus pattern recognition / Ajay Divakaran, Kadir A. Peker, Fu Chang // Conference: Image Processing, 2004. ICIP '04. 2004 International. – pp. 34. – Текст : непосредственный.

21. Francesco C. Machine Learning for Audio, Image and Video Analysis: Theory and Applications / Francesco Camastra, Alessandro Vinciarelli // Second Edition, Springer London Heidelberg, 2015 – pp. 559. – Текст : непосредственный.