МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования «Ярославский государственный университет им. П.Г. Демидова»

Кафедра теоретической информатики

Сдано на кафедру
«__» _____ 2023 г.
Заведующий кафедрой,
д. ф.-м. н., профессор
_____ Е. В. Кузьмин

Выпускная квалификационная работа

Алгоритм распознавания жестов рук на видео

по направлению подготовки 02.04.02 Фундаментальная информатика и информационные технологии

Научный руководитель
к. ф.-м. н., доцент
_____ Н. С. Лагутина
«__» _____ 2023 г.

Студент группы ИТ-21МО
_____ Э. В. Куликова
«__» _____ 2023 г.

Ярославль 2023 г.

Реферат

Объем 46 с., 5 гл., 22 рис., 18 табл., 16 источников.

Распознавание жестов, распознавание жестов на видео, классификация жестов, язык жестов, русский жестовый язык.

Цель работы – разработать алгоритм распознавания жестов рук на видео в реальном времени с использованием методов компьютерного зрения и машинного обучения.

В результате исследования был разработан алгоритм распознавания жестов рук на видео, который позволяет распознавать и классифицировать некоторые цифры и буквы русского жестового языка. Алгоритм был реализован на языке программирования Python с использованием библиотек OpenCV, MediPipe и TensorFlow. Кроме того, самостоятельно был собран корпус данных для русского жестового языка. Была проведена оценка качества алгоритма на основе анализа его производительности и посчитанных метрик на реальных видео.

Разработанный алгоритм распознавания жестов рук на видео может быть использован в различных областях, таких как контроль жестов для людей с ограниченными возможностями, управление умными устройствами и тому подобное. Дальнейшее развитие данного направления исследований может привести к созданию более эффективных методов распознавания жестов рук на видео.

Содержание

Введение

Распознавание жестов — сложная задача, которая уже давно находится в центре внимания исследований компьютерного зрения и машинного обучения. Распознавание жестов рук, в частности, становится все более важной темой в последние годы, поскольку оно имеет широкий спектр приложений в таких областях, как виртуальная и дополненная реальность и взаимодействие человека и робота.

Ещё важным применением распознавания жестов является помощь глухим, слабослышащим или немым людям. Для этих людей язык жестов часто является основным средством общения. Точно распознавая и интерпретируя жесты рук, технологии могут помочь преодолеть разрыв в общении между этими людьми и остальным обществом. Это может улучшить качество их жизни и повысить их способность участвовать в самых разных видах деятельности.

Поэтому одной из ключевых задач сейчас является возможность точного отслеживания и интерпретации жестов рук в режиме реального времени, особенно на видео. Существует несколько ключевых проблем связанных с распознаванием жестов рук в видеорядах: руки постоянно находятся в движении, меняют углы наклона и перекрывают друг друга, изменчивость формы и размеров рук. Всё это может затруднить создание универсального алгоритма.

Кроме того, жесты рук сильно зависят от контекста, а это означает, что один и тот же жест может иметь разное значение в зависимости от ситуации. Например, жест рукой, означающий «стоп» в одном контексте, может означать «подойти» в другом.

В этой работе проводятся исследования некоторых ключевых методов и алгоритмов, используемых для распознавания жестов рук на видео, а также будет рассмотрено их потенциальное применение в области русского жестового языка.

Задача по разработке системы распознавания жестов рук русского жестового языка на видео в реальном времени пока еще мало исследована, поэтому является особенно важной. Она может значительно улучшить взаимодействие между глухими и слышащими людьми. Такая система будет полезна в образовательных учреждениях, медицинских учреждениях, офисах и других местах, где глухие люди могут столкнуться с трудностями в общении.

Постановка задачи

В ходе данной выпускной квалификационной работы была поставлена задача: разработать и реализовать алгоритм распознавания цифр и букв русского и американского жестового языка на видео в реальном времени.

Поставленная задача включает себя следующие подзадачи:

- Самостоятельное создание корпуса данных с цифрами и буквами жестового русского языка;
- Подбор корпуса данных с цифрами и буквами жестового американского языка;
- Обработка наборов данных для получения дополнительных характеристик, которые могут быть использованы для классификации цифр и букв;
- Проведение экспериментов по классификации цифр и букв русского и американского жестового языка;
- Определение качества работы разработанного алгоритма.

1. Описание предметной области

1.1. Язык жестов

В настоящее время более 5% населения мира (около 430 миллионов человек) нуждаются в реабилитации для решения проблемы инвалидизирующей потери слуха. Потеря слуха более широко распространена среди людей в возрасте старше 60 лет. Однако, и около 1,1 миллиарда молодых людей подвергаются риску потери слуха из-за прослушивания музыки на слишком высоком, опасном для слуха уровне громкости. Согласно прогнозам, к 2050 г. почти 2,5 миллиарда человек будут страдать от проблем со слухом в той или иной степени [1]. Обеспечение возможности общения этих людей в обществе является важной социальной задачей, чтобы устранить психологические, идеологические и межкультурные барьеры.

Язык жестов является основным средством коммуникации с глухими или слабослышащими людьми. Это естественный язык, на котором общаются с помощью жестов, а не звуков. Он используется для передачи информации и выражения эмоций и чувств. Жестовый язык включает в себя различные жесты, каждый из которых имеет свое значение. Эти жесты могут использоваться для передачи слов и фраз, а также для выражения концепций и идей. Даже в пределах одной языковой группы существуют значительные различия, поэтому разработка системы преобразования языка жестов в речь для улучшения взаимодействия между людьми является острой необходимостью. К тому же, в разных регионах и странах есть свои языки жестов, из-за этого могут возникнуть коммуникативные трудности, в особенности, у тех, кто не знает местного языка жестов.

Русский жестовый язык изучен недостаточно хорошо, использование знаков варьируется от региона к региону. Помимо этого, не хватает общедоступного набора жестового русского языка для исследований в области компьютерного зрения и машинного обучения, поэтому его необходимо собирать самостоятельно.

1.2. Методологии распознавания языка жестов

В связи с развитием компьютерных технологий многие исследователи разработали новые методы для помощи людям, имеющим ограниченные физические возможности. Необходимость систем машинного перевода обусловлена нехваткой человеческих ресурсов в области сурдоперевода и в ситуациях, когда посредничество в общении между глухими или слабослышащими и слышащими людьми не всегда

желательно, например, в сфере медицины (конфиденциальность данных или врачебная тайна).

Существует два типа распознавания жестов рук: на основе носимых перчаток и на основе машинного зрения. Минус первого метода в том, что он дорог и требует ношения на руке специального устройства для распознавания жестов, а также нестабилен в некоторых средах. Второй метод основан на обработке изображений, где последовательность операций выполняется следующим образом: захват изображения с помощью веб-камеры, сегментация, извлечение признаков и классификация жестов [2].

Сообщества с нарушениями слуха часто используют язык жестов, который представляет собой систему, использующую визуально-мануальную модальность для передачи смысла. Язык жестов зависит в основном от жестов рук, движений тела и выражения лица. Распознавание языка жестов (SLR) — сложная задача, особенно распознавание динамических знаков, зависящих от движения. Вот почему многие исследователи заинтересованы в разработке приложения SLR с целью уменьшения барьера между сообществом с нарушениями слуха и обществом [3].

Методики распознавания языка жестов на основе компьютерного зрения принято делить на две категории: статические и динамические. Статические признаки — это те, которые требуют обработки только одного изображения на входе классификатора, то есть его можно рассматривать как картинку формы руки. Динамические знаки можно рассматривать как видео, содержащее ряд последовательных кадров для построения знака. Как правило, в языке жестов знаки строятся из серии быстрых движений рук и тела. Следовательно, статическое распознавание не является хорошим решением проблем языка жестов, поскольку оно не может справиться с вариациями знаков. Значит, динамическое решение является более эффективным и действенным [3].

Есть некоторые проблемы, с которыми сталкивается SLR, которые можно классифицировать как основные и вторичные факторы.

1. Основные факторы:
   - Форма руки: разница в форме руки меняет знак;
   - Расположение руки: расположение руки относительно тела может изменить значение знака, даже если форма руки такая же;
   - Движение руки: самый сложный параметр, так как знак может содержать набор движения с разными направлениями и формами.

2. Второстепенные факторы:

- Выражение лица: выражение лица играет жизненно важную роль в иллюстрации знака; повышает смысл и силу смысла в процессе общения;
- Ориентация ладони: направление ладони при выполнении знака — вверх или вниз, вправо или влево [3].

Многие исследователи использовали методы извлечения признаков вручную с алгоритмами машинного обучения для классификации жестов рук, но в последнее время в большинстве разработок использовались методы глубокого обучения. Изначально для распознавания жестов рук применялись свёрточные нейронные сети (CNN), но с помощью них было трудно распознать динамические жесты рук, содержащие пространственно-временную информацию. Некоторые исследователи использовали рекуррентные нейронные сети (RNN), которые в основном похожи на CNN, хотя свёрточные нейронные сети оказались более успешными. Недавно разработчики использовали долговременную кратковременную память (LSTM) для извлечения долговременной зависимости. Комбинация CNN и LSTM использовалась для достижения высокой точности распознавания жестов рук. Долгосрочная зависимость требует высокой сложности вычислений, что является основной проблемой LSTM. Нейронные сети, основанные на внимании, напротив, создают кратковременную зависимость, которая требует меньшей вычислительной сложности [2].

Таким образом, учитывая все особенности и сложности распознавания жестов рук на видео в реальном времени, было решено провести несколько экспериментов с разными моделями нейронных сетей для извлечения признаков и сравнить их результаты.

1.3. Обзор аналогичных работ

В настоящий момент существует несколько алгоритмов по распознаванию жестов рук. Например, в статье [4] представлена реализация модели рекуррентной нейронной сети (RNN) с использованием блоков долговременной кратковременной памяти (LSTM) и плотных слоев для разработки классификатора жестов для управления протезами кисти. В качестве набора данных взяты электромиографические (ЭМГ) сигналы (ЭМГ-сигналы – разность потенциалов, возникающая в мышцах человека в покое и при их активации) пяти жестов. Каждый жест записывался в течение 20 секунд с помощью специальной ЭМГ-повязки. Для каждого жеста было записано 20 000 образцов. Результаты, полученные с помощью

предложенной модели, были протестированы на реальных видео, где средняя F-мера (мера точности теста) для 5 классов составила 0.8509.

В работе [5] предлагается модель множественного параллельного потока 2D CNN (двумерная свёрточная нейронная сеть) для распознавания поз рук. Предлагаемая модель включает в себя несколько этапов и слоев для определения положения рук по картам изображений, полученных на основе данных глубины. В качестве наборов данных берутся три общедоступных эталонных набора Kaggle (10 классов), First Person (9 классов) и Dexter (7 классов), где количество данных для обучения составляет – 13 375, 98 842 и 19 519 кадров. Средняя F-мера предлагаемого метода составляет 1, 1 и 0.92 при использовании набора данных о положении рук Kaggle, First Person и Dexter соответственно.

В статье [6] рассматриваются подходы к распознаванию жестовых языков глухих на примере русского жестового языка (РЖЯ). Авторами был собран собственный корпус данных для РЖЯ, который включает 35 000 жестов (изображения и 10 000 видеофайлов), построена модель рекуррентной сети (LSTM). Средняя точность правильного распознавания жестов проверялась на реальных видео, где добровольцы показывали предложения с помощью жестов РЖЯ, и она составила 0.95.

Все вышеописанные модели довольно хорошо распознают жесты рук, как на статичных изображениях, так и на видео. Каждая модель обучена на больших наборах данных, что позволило им выявить более сложные закономерности в движениях рук, поэтому модели достаточно результативны и точны в распознавании жестов рук.

Одним из главных минусов алгоритмов распознавания жестов рук является необходимость большого и разнообразного набора данных. Две статьи использовали корпуса данных, содержащие от 5 до 10 жестов рук. Более сложные жесты могут потребовать дополнительных данных и времени для обучения алгоритма. Кроме того, эти алгоритмы не всегда могут правильно распознавать жесты, выполненные в разных условиях, таких как разное освещение, фон и т.д. Также в случае модели рекуррентной нейронной сети с использованием блоков долговременной кратковременной памяти (LSTM) и плотных слоев [4], недостатком является то, что она требует использования специальной ЭМГ-повязки для записи данных, что делает ее менее удобной для использования на практике.

2. Сбор корпусов данных

2.1. Методология

Задача сбора корпуса данных, содержащего видео и фотографии, является важной задачей для многих исследований в области компьютерного зрения и машинного обучения. Ниже будет представлена одна из методологий, которая использовалась при самостоятельном создании набора данных цифр и букв русского жестового языка.

Методология включает в себя следующие шаги:

- Определение цели и задач. Первым шагом при сборе корпуса данных является определение цели и задач исследования. Это поможет определить, какие типы видео и фотографий должны быть включены в корпус данных. Например, если исследование связано с распознаванием объектов на фотографиях, то необходимо включить фотографии, на которых присутствуют различные объекты.
- Определение источников данных. Источники могут быть различными, такими как интернет, социальные сети, базы данных и т.д. Необходимо определить, какие источники будут использоваться для сбора данных, и какие ограничения могут быть наложены на использование этих источников.
- Сбор данных. Третьим шагом является непосредственный сбор данных. Для сбора фотографий может быть использовано программное обеспечение для сканирования сайтов или социальных сетей, а для сбора видео может быть использовано программное обеспечение для загрузки видео с любых видеохостингов. Кроме того, можно найти добровольцев и с помощью них заснять видео или фото самостоятельно. При сборе данных необходимо учитывать правовые и этические аспекты, такие как защита личных данных и авторские права.
- Обработка данных. Это может включать в себя удаление дубликатов, удаление нежелательных изображений или видео, а также ручную проверку данных на наличие ошибок.
- Аннотация данных. Это может включать в себя маркировку изображений или видео, а также создание метаданных для каждого элемента данных.
- Разбиение данных на обучающую и тестовую выборку. Обучающая выборка используется для обучения модели, а тестовая выборка - для проверки точности модели.

– Оценка качества данных. Это может включать в себя проверку точности разметки данных, а также оценку покрытия данных для каждого класса объектов.

2.2. Корпус данных для русского жестового языка

Поскольку не существует общедоступного и общепринятого набора данных для цифр и букв русского жестового языка, то он был создан самостоятельно с помощью добровольцев. При сборе корпуса данных использовались принципы методологии, описанной выше.

Сбор корпуса данных для русского жестового языка включал в себя несколько этапов:

1. Цифры. На первом этапе записывались на видео цифры русского жестового языка. Участвовало 19 добровольцев, из которых 10 женщин и 9 мужчин в возрасте от 20 до 55 лет. Добровольцы показывали цифры от 1 до 10 на правой и левой руке с помощью жестов. Инструкции о том, как показывать цифры жестового русского языка были взяты из проекта «Словарь. Русский жестовый язык» (https://surdo.me). Было записано 38 видео и взято одно видео с цифрами от 1 до 10 из проекта «Словарь. Русский жестовый язык». Дополнительно было снято 2 видео с правой и левой рукой для проверки на реальных видео качества распознавания и классификации жестов цифр.
2. Буквы. На втором этапе записывались на видео буквы русского жестового языка. Участвовало 11 добровольцев, из которых 7 женщин и 4 мужчин в возрасте от 20 до 55 лет. Добровольцы показывали русский алфавит с помощью жестов. Инструкции о том, как показывать буквы жестового русского языка были взяты из проекта «Словарь. Русский жестовый язык» (https://surdo.me). Было записано 10 видео и взято одно видео из проекта «Словарь. Русский жестовый язык». Дополнительно было снято видео для проверки на реальном видео качества распознавания и классификации жестов букв.
3. На третьем этапе все видео были подвергнуты раскадровке и вручную отобраны лучшие фотографии, где чётко видно жест и он показан достоверно. Для последующей классификации жесты были разбиты на классы. Для цифр классы с наименованием от 1 до 10, которые соответствуют цифрам, и 25 классов для букв с их наименованием соответственно (взяты более статичные жесты).

В результате всех вышеуказанных действий, получилось 10 классов с жестами цифр от 1 до 10 по 420 цветных фотографий на каждый жест и 25 классов с жестами букв с N цветных фотографий на каждый жест (показ букв более динамичный, поэтому каждый участник показывал жест N количество секунд).

2.3. Корпус данных для американского жестового языка

Для американского жестового языка (ASL) существует достаточное количество общедоступных наборов данных. Поэтому было выбрано два набора данных, один для классификации цифр, другой для – букв, которые соответствовали целям и задачам исследования. Оба корпуса данных были найдены на «Kaggle» – система организации конкурсов по исследованию данных, а также социальная сеть для специалистов по обработке данных и машинному обучению [7].

Первый набор данных – это набор цифр от 0 до 9 американского языка жестов (https://www.kaggle.com/datasets/rayeed045/american-sign-language-digit-dataset). Содержит 10 классов по 500 цветных изображений рук на каждый жест с тёмным фоном с разрешением 400 × 400 (пример на рис. 1).

Второй набор данных – это набор букв от a до z американского алфавита (https://www.kaggle.com/datasets/kapillondhe/american-sign-language/code). Из корпуса данных было выбрано 3000 цветных фотографий жестов с разрешением 400 × 400 для каждой буквы (пример на рис. 2).

Поскольку найденные корпуса данных не содержали видео, было дополнительно снято два видео, на которых можно проверить качество классификации цифр и букв на реальном видео.

Рис. 1 - Пример набора данных цифр ASL

Рис. 2 - Пример набора данных букв ASL

3. Обработка корпусов данных

3.1. MediaPipe

Для обработки собранных наборов данных использовалось решение «MediaPipe Hand Landmarker». Оно является частью проекта «MediaPipe» с открытым исходным кодом, код решения которого можно дополнительно настроить в соответствии с потребностями разработчика. «MediaPipe», в свою очередь, – это фреймворк с открытым исходным кодом, представленный Google, который помогает создавать мультимодальные конвейеры машинного обучения [8].

Задача «MediaPipe Hand Landmarker» позволяет обнаружить ориентиры рук на изображении. Его можно использовать для локализации ключевых точек рук и визуализации ориентиров. Это решение работает с данными изображения с помощью модели машинного обучения в виде статических данных или непрерывного потока и выводит ориентиры рук в координатах изображения, ориентиры рук в мировых координатах и принадлежность (левая/правая рука) нескольких обнаруженных рук [8].

«Hand Landmarker» использует пакет моделей с двумя упакованными моделями: моделью обнаружения ладони и моделью обнаружения ориентиров руки. Модель обнаружения ладони определяет местоположение рук на входном изображении, а модель обнаружения ориентиров для рук определяет конкретные ориентиры для рук на обрезанном изображении руки, определенном моделью обнаружения ладони [8].

Пучок моделей ориентиров руки определяет локализацию ключевых точек 21 координаты костяшек кисти в пределах обнаруженных областей руки. Модель была обучена примерно на 30 тыс. реальных изображений, а также на нескольких синтетических моделях рук, наложенных на различные фоны [8]. Определение 21 ориентира приведено на рис. 3.

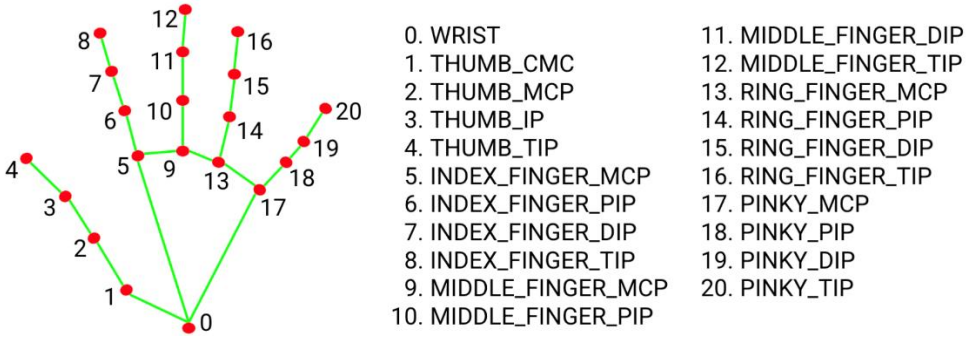

Рис. 3 - 21 ключевая точка руки

В данной работе решение «Hand Landmarker» использовалась как для обнаружения рук на видео, так и для получения ключевых точек обнаруженных рук. Ключевые точки рук будут использоваться для классификации жестов цифр и букв русского и американского жестового языка.

3.2. Корпуса данных для русского жестового языка

Оба корпуса данных с цифрами и буквами русского жестового языка (RSL) были переработаны решением «MediaPipe Hand Landmarker». Было решено разделить каждый набор данных на два.

Один содержит документы в формате JSON (рис. 4), где каждая фотография набора представляет собой следующую информацию: сколько рук обнаружено на фото и массив данных длинной 43 элемента. Массив содержит ключевые точки обнаруженных рук в формате x, y и z: 21 ориентир для правой руки, 21 ориентир для левой руки и точка соотношения положений двух рук относительно друг друга. Если на фотографии обнаружена одна рука, то элементы с индексами после 21 будут нулевыми. Все координаты также были нормализованы от 0 до 1 относительно координат изображения.


```
{
  "num_hands": 2,
  "landmarks": [
    [
      1,
      0.9376944347895629,
      -1.6562921700824518e-7
    ],
    [
      0.8552957420378341,
      0.8466151352315685,
      -0.0012534725246950984
    ],
    [
      0.6261349278546047,
      0.7566050155486314,
      -0.0054916092194616795
    ],
    [
      0.4559098215631299,
      0.68231147092206,
      -0.009846813045442104
    ],
    [
      0.3920729425002743,
      0.5881747352426417,
      -0.013850336894392967
    ],
    ...
    [
      -0.29899818204940826,
      0.3388255268253013,
      -0.06701805721968412
    ]
  ]
}
```


Рис. 4 - Пример экземпляра набора данных с точками

Второй набор – изображения, содержащие кисти рук, для каждого жеста с разрешением 64 × 64 (рис. 5 и рис. 6).

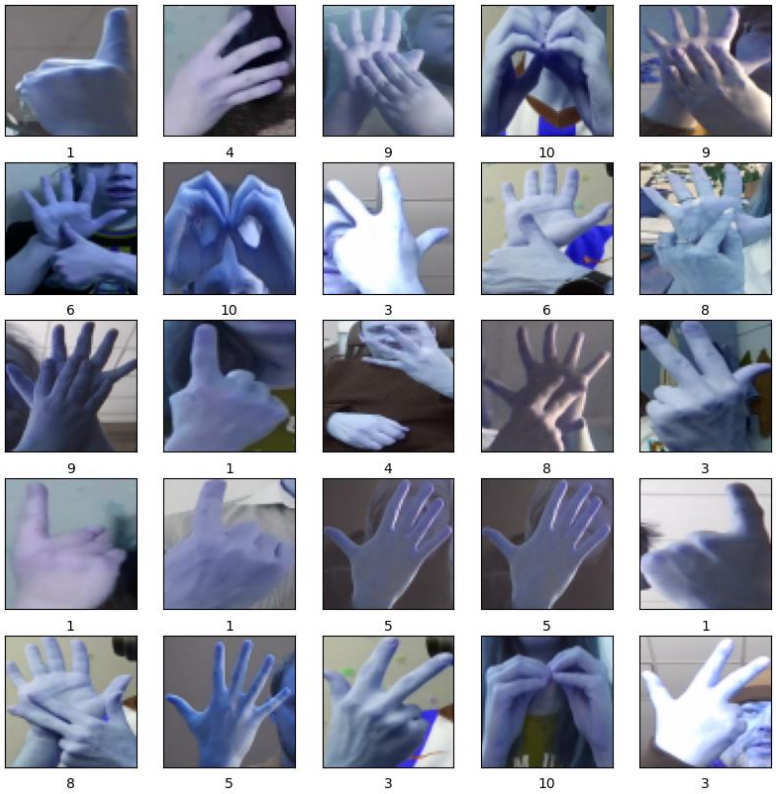

Рис. 5 - Пример набора данных цифр RSL

Рис. 6 - Пример набора данных букв RSL

Для цифр русского жестового языка теперь существует два корпуса данных: 10 классов по 300 документов в формате JSON для каждого жеста и 10 классов по 300 цветных фотографий для каждого жеста только обнаруженных рук с разрешением 64 × 64. А для букв русского жестового языка - 25 классов по 8 документов в формате JSON для каждого жеста, которые содержат информацию о 50 кадрах, в течение которых, показывался жест, то есть массивы размером 50 × 43 ориентиров рук. И 25 классов по 8 папок для каждого человека с 50 цветными фотографиями для каждого жеста только обнаруженных рук разрешением 64 × 64. Если жест показывался меньше чем за 50 кадров, то в случае точек недостающие данные дополнялись методом интерполяции, а в случае изображений дополнялись копией предыдущего.

3.3. Корпуса данных для американского жестового языка

Оба корпуса данных с цифрами и буквами американского жестового языка также были переработаны решением «MediaPipe Hand Landmarker». Поскольку оба набора и так содержали фотографии только рук, то в данном случае, решение MediaPipe использовалось для извлечения ключевых точек. Таким образом, появилось два дополнительных корпуса данных. Для цифр - 10 классов по 430 документов в формате JSON, для букв - 26 классов по 3000 документов в формате JSON для каждого жеста соответственно. Все массивы данных также были нормализованы от 0 до 1 относительно координат изображения.

4. Алгоритм распознавания языка жестов

4.1. Модели нейронных сетей

Нейронные сети - это математические модели, состоящие из множества связанных между собой нейронов, которые обрабатывают информацию и передают ее дальше по сети. Они, в основном, используются для анализа и обработки больших объёмов данных, что позволяет получать более точные прогнозы и принимать более обоснованные решения [9].

Собранные корпуса данных для русского и американского жестового языка будут использоваться для обучения моделей нейронных сетей, главной задачей которых является научиться правильно классифицировать цифры и буквы языков жестов. Были выбраны четыре модели: для распознавания жеста по ключевым точкам руки – многослойный персептрон (MLP), одномерная свёрточная (1D CNN) и сеть долгой краткосрочной памяти (LSTM), а также по фото – двумерная свёрточная сеть (2D CNN).

Многослойный персептрон (MLP) – это тип нейронной сети, которая состоит из одного входного слоя, одного или нескольких скрытых слоев и одного выходного слоя. Сигнал проходит от входного слоя через скрытые слои к выходному слою, при этом связь допускается только между соседними слоями. Модель используется для решения различных задач, таких как прогнозирование, классификация и распознавание образов [10].

Одномерная свёрточная нейронная сеть (1D CNN) – это тип нейронной сети, которая обычно используется для задач обработки сигналов, таких как классификация изображений или аудио. Она предназначена для извлечения признаков из одномерных последовательностей данных, таких как временные ряды или сигналы [11].

Нейронная сеть долгой краткосрочной памяти (LSTM) – это тип рекуррентной нейронной сети (RNN), которая особенно полезна для обработки и прогнозирования последовательных данных, таких как временные ряды или текст на естественном языке. Во время обучения используется последовательная информация, которая проходит через нейронную сеть от входного вектора к выходным нейронам, а ошибка вычисляется и распространяется обратно через сеть для обновления параметров сети. Информация в этих сетях включает петли в скрытый слой, позволяя информации течь разнонаправленно, так что скрытое состояние означает прошлую информацию, имеющуюся на данном временном шаге [12].

Двумерная свёрточная нейронная сеть (2D CNN) - это тип архитектуры глубокого обучения, который наиболее широко используется для обработки изображений и временных рядов. Она подходит для анализа и обработки двумерных данных, поскольку признаки извлекаются из входных данных посредством свёртки. Модель состоит из повторяющегося свёрточного слоя, объединяющего слоя и полносвязанного слоя [13].

Многослойная полносвязная сеть была выбрана, поскольку является классической сетью, которая годится для многих задач классификации. Последние три модели – исходя из применения их исследователями в похожих областях.

4.2. Алгоритм

Для работы с видео используется библиотека с открытым кодом алгоритмов компьютерного зрения, обработки изображений и численных алгоритмов общего назначения – OpenCV. Она предоставляет очень простой интерфейс для захвата видео с камеры, в данном случае, встроенной веб-камеры на ноутбуке.

Для распознавания рук на видео используется решение «MediaPipe Hand Landmarker», которое локализует на кадре ключевые точки правой и/или левой руки и может визуализировать их (пример на рис. 7).

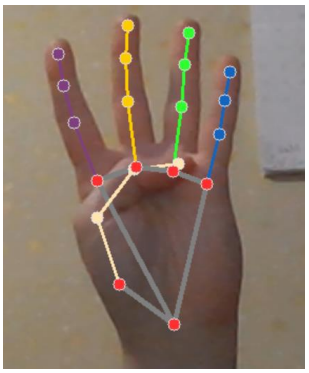

Рис. 7 - Пример работы "MediaPipe Hand Landmarker"

Алгоритм распознавания языка жестов включает в себя следующие шаги:

1. Подключение решения «MediaPipe Hand Landmarker» в режиме одиночного обнаружения (каждый кадр видео рассматриваем отдельно) и минимальным значением достоверности – 0.6 (то есть при оценке >= 0.6 обнаружение рук считается успешным);
2. Захват живого потока с веб-камеры;
3. Первичная обработка каждого кадра видео для распознавания руки:
   - Прочитать кадр;

- Перевернуть кадр вокруг оси Y для правильного вывода рук;
- Преобразовать изображение BGR в RGB и передать его на обработку в конвейер решения «MediaPipe Hand Landmarker», где кадр пройдёт через модель обнаружения ладони, а потом пройдёт через модель ориентира руки. В результате, получим 21 ориентир для каждой обнаруженной руки на данном видеокадре;

4. Вторичная обработка каждого кадра видео для классификации жеста:
   - Если выбрана модель двумерной свёрточной сети. Кадр обрезается с учётом найденных на прошлом шаге ориентиров обнаруженных рук, размер изображения меняется на 64 × 64, кадр нормализуется и отправляется на вход обученной 2D CNN, где происходит предсказание с использованием обученной нейронной сети;
   - Если выбрана одна из моделей, обученная на ключевых точках рук (MLP/1D CNN/LSTM). Представить кадр как массив из 43 точек, где 21 ориентир для правой руки, 21 ориентир для левой руки, вычисленные на прошлом шаге, и посчитанная точка соотношения положений двух рук относительно друг друга с помощью алгоритма Евклида. Если на кадре была обнаружена одна рука, то элементы с индексами после 21 будут нулевыми. Далее координаты нормализуются от 0 до 1 и массив данных отправляется на вход обученной сети, где она по ключевым точкам рук пытается предсказать какой жест изображён на кадре;
5. На кадре визуализируются ориентиры, обнаруженных рук, и отображается символ, соответствующий жесту (пример на рис. 8);

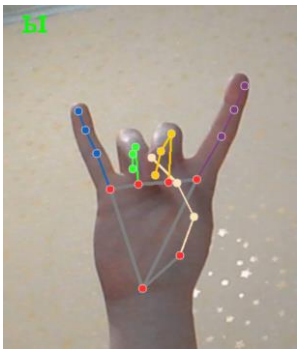

Рис. 8 - Пример распознавания руки и классификации жеста

6. Шаги 3-5 повторяются до тех пор, пока пользователь не прекратит использовать режим распознавания;

7. Отпустить захват видеопотока и освободить ресурсы.

На основе данного алгоритма было создано приложение, распознающее жестовый язык. Входными данными для приложения является видеопоток, захваченный веб-камерой. В качестве выходных данных пользователь может получить видеоизображение с расшифровкой в виде текстового предсказания распознанного жеста. Приложение реализовано на языке программирования Python с использованием библиотек OpenCV, MediPipe, TensorFlow и разработанного алгоритма распознавания жестов на основе обученной ранее нейронной сети.

5. Эксперименты

Для эффективной классификации жестов рук было решено создать четыре модели нейронных сетей: многослойный персептрон, одномерная свёрточная сеть, сеть долгой краткосрочной памяти для обучения их на входных данных, которые представляют собой ключевые точки рук, и двумерная свёрточная сеть для обучения на фотографиях жестов. Во время экспериментов проводились исследования по влиянию архитектуры и параметров обучения на качество работы нейронных сетей. Были протестированы различные комбинации архитектур и параметров, включая количество слоев, количество нейронов в каждом слое, функции активации, скорость обучения и размер пакета данных.

Нейронные сети обучались на корпусах данных, описанных в разделах 3.2 и 3.3. Каждый корпус данных был разделён на три выборки: обучающую (60%), валидационную (20%) и тестовую (20%). Обучение проводилось на обучающей выборке, а результаты оценивались на валидационной и тестовой выборках.

Эксперименты проводились в три этапа:

1. Первый этап включал в себя создание таких моделей сетей, которые давали наилучший результат классификации цифр русского и американского жестового языка. Данные для обучения:
   - Для MLP, 1D CNN и LSTML:
     - ключевые точки рук цифр русского жестового языка (1800 экземпляров);
     - ключевые точки рук цифр американского жестового языка (2580 экземпляров);
   - Для 2D CNN:
     - фотографии рук цифр русского жестового языка (1800 экземпляров);
     - фотографии рук цифр американского жестового языка (3000 экземпляров);
2. Второй этап включал в себя создание таких моделей сетей, которые давали наилучший результат классификации букв русского и американского жестового языка. Данные для обучения:
   - Для MLP, 1D CNN и LSTML:
     - ключевые точки рук букв русского жестового языка (6000 экземпляров);

        - ключевые точки рук букв американского жестового языка (46800 экземпляров);
    - Для 2D CNN:
        - фотографии рук букв русского жестового языка (6000 экземпляров);
        - фотографии рук букв американского жестового языка (46800 экземпляров);
3. Третий этап включал в себя создание таких моделей сетей, которые давали наилучший результат классификации цифр и букв русского и американского жестового языка. Данные для обучения:
    - Для MLP, 1D CNN и LSTML:
        - ключевые точки рук цифр и букв русского жестового языка (7800 экземпляров);
        - ключевые точки рук цифр и букв американского жестового языка (49310 экземпляров);
    - Для 2D CNN:
        - фотографии рук цифр и букв русского жестового языка (7800 экземпляров);
        - фотографии рук цифр и букв американского жестового языка (49800 экземпляров).

5.1. Архитектуры нейронных сетей

При создании архитектуры любой нейронной сети следует руководствоваться следующими шагами:

1. Определение задачи, которую должна решать сеть;
2. Сбор и подготовка данных для обучения сети;
3. Выбор типа сети, соответствующего задаче;
4. Определение количества слоев и нейронов в каждом слое;
5. Определение функции активации для каждого слоя;
6. Компиляция и обучение модели;
7. Тестирование и настройка гиперпараметров.

Первые три шага были выполнены и описаны в разделах 2 – 4.

При определении количества слоев и нейронов в каждом слое, учитывается сложность задачи и доступные ресурсы. Поскольку экземпляров для обучения недостаточно много, в особенности для русского жестового языка, то все модели содержат не более шести слоёв и каждый слой содержит не более 256 нейронов.

При выборе функций активации необходимо учитывать тип задачи и тип данных. Так как поставленная задача включает в себя обработку изображений и многоклассовую классификацию, то были выбраны такие функции активации как ReLu и Softmax. ReLU (Rectified Linear Unit) - это функция, которая пропускает все положительные значения, а все отрицательные значения заменяет на 0. Softmax - это функция, которая преобразует входные данные в вероятности, сумма которых равна 1 [14].

Для сетей MLP, 1D CNN и LSTM входными данными являются ключевые точки руки, то есть массив размером 43 × 3, а для 2D CNN форма входных данных будет – (64, 64, 3), что соответствует трёхканальному изображению размером 64 × 64.

Для задачи классификации жестов выходными параметрами будут вероятности каждого класса, а количество этих параметров определяется количеством классов: для цифр – 10, для букв русского жестового языка – 25, для букв американского жестового языка – 26, для цифр и букв – 35 или 36 в зависимости от языка жеста.

В итоге, на рисунках 9 – 20 показаны разработанные оптимальные архитектуры на каждом этапе экспериментов.

1. Классификация жестов цифр:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_4 (Dense)             (None, 43, 129)           516

 flatten_1 (Flatten)         (None, 5547)              0

 dense_5 (Dense)             (None, 128)               710144

 dense_6 (Dense)             (None, 256)               33024

 dense_7 (Dense)             (None, 10)                2570

=================================================================
Total params: 746,254
Trainable params: 746,254
Non-trainable params: 0
```

Рис. 9 - Архитектура MLP для классификации жестов цифр

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d_1 (Conv1D)           (None, 42, 64)            448

 dense_2 (Dense)             (None, 42, 16)            1040

 max_pooling1d_1 (MaxPooling  (None, 21, 16)           0
 1D)

 flatten_1 (Flatten)         (None, 336)               0

 dense_3 (Dense)             (None, 10)                3370

=================================================================
Total params: 4,858
Trainable params: 4,858
Non-trainable params: 0
```

Рис. 10 - Архитектура 1D CNN для классификации жестов цифр

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm (LSTM)                 (None, 43, 64)            17408

 lstm_1 (LSTM)               (None, 43, 128)           98816

 lstm_2 (LSTM)               (None, 64)                49408

 dense (Dense)               (None, 64)                4160

 dense_1 (Dense)             (None, 32)                2080

 dense_2 (Dense)             (None, 10)                330

=================================================================
Total params: 172,202
Trainable params: 172,202
Non-trainable params: 0
```

Рис. 11 - Архитектура LSTM для классификации жестов цифр

```
Model: "sequential_3"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_18 (Conv2D)          (None, 64, 64, 32)        896

 conv2d_19 (Conv2D)          (None, 62, 62, 32)        9248

 max_pooling2d_9 (MaxPooling  (None, 31, 31, 32)       0
 2D)

 dropout_12 (Dropout)        (None, 31, 31, 32)        0

 conv2d_20 (Conv2D)          (None, 31, 31, 64)        18496

 conv2d_21 (Conv2D)          (None, 29, 29, 64)        36928

 max_pooling2d_10 (MaxPoolin  (None, 14, 14, 64)       0
 g2D)

 dropout_13 (Dropout)        (None, 14, 14, 64)        0

 conv2d_22 (Conv2D)          (None, 14, 14, 64)        36928

 conv2d_23 (Conv2D)          (None, 12, 12, 64)        36928

 max_pooling2d_11 (MaxPoolin  (None, 6, 6, 64)         0
 g2D)

 dropout_14 (Dropout)        (None, 6, 6, 64)          0

 flatten_3 (Flatten)         (None, 2304)              0

 dense_6 (Dense)             (None, 512)               1180160

 dropout_15 (Dropout)        (None, 512)               0

 dense_7 (Dense)             (None, 10)                5130

=================================================================
Total params: 1,324,714
Trainable params: 1,324,714
Non-trainable params: 0
```

Рис. 12 - Архитектура 2D CNN для классификации жестов цифр

2. Классификация жестов букв:

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 43, 129)           516

 flatten (Flatten)           (None, 5547)              0

 dense_1 (Dense)             (None, 128)               710144

 dense_2 (Dense)             (None, 64)                8256

 dense_3 (Dense)             (None, 25)                1625

=================================================================
Total params: 720,541
Trainable params: 720,541
Non-trainable params: 0
```

Рис. 13 - Архитектура MLP для классификации жестов букв

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 42, 64)            448

 dense (Dense)               (None, 42, 16)            1040

 max_pooling1d (MaxPooling1D  (None, 21, 16)           0
 )

 flatten (Flatten)           (None, 336)               0

 dense_1 (Dense)             (None, 25)                8425

=================================================================
Total params: 9,913
Trainable params: 9,913
Non-trainable params: 0
```

Рис. 14 - Архитектура 1D CNN для классификации жестов букв

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_3 (LSTM)               (None, 43, 64)            17408

 lstm_4 (LSTM)               (None, 43, 128)           98816

 lstm_5 (LSTM)               (None, 64)                49408

 dense_3 (Dense)             (None, 64)                4160

 dense_4 (Dense)             (None, 32)                2080

 dense_5 (Dense)             (None, 25)                825

=================================================================
Total params: 172,697
Trainable params: 172,697
Non-trainable params: 0
```

Рис. 15 - Архитектура LSTM для классификации жестов букв

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 32)        896

 conv2d_1 (Conv2D)           (None, 62, 62, 32)        9248

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 dropout (Dropout)           (None, 31, 31, 32)        0

 conv2d_2 (Conv2D)           (None, 31, 31, 64)        18496

 conv2d_3 (Conv2D)           (None, 29, 29, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 14, 14, 64)        0

 conv2d_4 (Conv2D)           (None, 14, 14, 64)        36928

 conv2d_5 (Conv2D)           (None, 12, 12, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 dropout_2 (Dropout)         (None, 6, 6, 64)          0

 flatten (Flatten)           (None, 2304)              0

 dense (Dense)               (None, 512)               1180160

 dropout_3 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 25)                12825

=================================================================
Total params: 1,332,409
Trainable params: 1,332,409
Non-trainable params: 0
```

Рис. 16 - Архитектура 2D CNN для классификации жестов букв

3. Классификация цифр и букв:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_5 (Dense)             (None, 43, 129)           516

 flatten_1 (Flatten)         (None, 5547)              0

 dense_6 (Dense)             (None, 256)               1420288

 dense_7 (Dense)             (None, 128)               32896

 dense_8 (Dense)             (None, 35)                4515

=================================================================
Total params: 1,458,215
Trainable params: 1,458,215
Non-trainable params: 0
```

Рис. 17 - Архитектура MLP для классификации жестов цифр и букв

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv1d (Conv1D)             (None, 42, 64)            448

 dense (Dense)               (None, 42, 32)            2080

 max_pooling1d (MaxPooling1D  (None, 21, 32)           0
 )

 flatten (Flatten)           (None, 672)               0

 dense_1 (Dense)             (None, 64)                43072

 dense_2 (Dense)             (None, 35)                2275

=================================================================
Total params: 47,875
Trainable params: 47,875
Non-trainable params: 0
```

Рис. 18 - Архитектура 1D CNN для классификации жестов цифр и букв

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 lstm_2 (LSTM)               (None, 43, 64)            17408

 lstm_3 (LSTM)               (None, 64)                33024

 dense_3 (Dense)             (None, 64)                4160

 dense_4 (Dense)             (None, 32)                2080

 dense_5 (Dense)             (None, 35)                1155

=================================================================
Total params: 57,827
Trainable params: 57,827
Non-trainable params: 0
```

Рис. 19 - Архитектура LSTM для классификации жестов цифр и букв

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 64, 64, 32)        896

 conv2d_1 (Conv2D)           (None, 62, 62, 32)        9248

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 dropout (Dropout)           (None, 31, 31, 32)        0

 conv2d_2 (Conv2D)           (None, 31, 31, 64)        18496

 conv2d_3 (Conv2D)           (None, 29, 29, 64)        36928

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 14, 14, 64)        0

 conv2d_4 (Conv2D)           (None, 14, 14, 64)        36928

 conv2d_5 (Conv2D)           (None, 12, 12, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 6, 6, 64)         0
 2D)

 dropout_2 (Dropout)         (None, 6, 6, 64)          0

 flatten (Flatten)           (None, 2304)              0

 dense (Dense)               (None, 512)               1180160

 dropout_3 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 35)                17955

=================================================================
Total params: 1,337,539
Trainable params: 1,337,539
Non-trainable params: 0
```

Рис. 20 - Архитектура 2D CNN для классификации жестов цифр и букв

5.2. Обучение нейронных сетей

После создания архитектуры нейронной сети ее необходимо обучить на наборе данных, чтобы она могла решать задачу, для которой была создана. Обучение нейронной сети включает в себя следующие шаги:

1. Подготовка данных – набор данных разбивается на обучающий, валидационный и тестовый наборы. Обучающий набор используется для обучения модели, валидационный набор используется для настройки гиперпараметров модели, а тестовый набор используется для оценки качества модели после ее обучения;
2. Компиляция модели – определяются функции потерь, метрики и оптимизатор, которые будут использоваться при обучении модели;
3. Обучение модели – модель обучается на обучающем наборе данных путем итеративного прохода через набор данных и

корректировки весов модели с помощью оптимизатора. В процессе обучения модели также используется валидационный набор данных для настройки гиперпараметров модели;

4. Оценка модели – после обучения модели она оценивается на тестовом наборе данных, чтобы определить ее точность и эффективность.

Перед обучением каждой нейронной сети, содержание каждого в отдельности корпуса данных было перемешено случайным образом и разделено на три выборки обучающуюся, валидационную и тестовую в процентном соотношении 60%, 20% и 20%. Метки классов были преобразованы в категориальный формат. Категориальный формат меток позволяет модели правильно интерпретировать метки классов и корректно обрабатывать их в процессе обучения.

В качестве функции потерь была выбрана категориальная кросс-энтропия, поскольку модели предназначены для классификации, когда классы представлены в виде категорий. Она измеряет расстояние между истинными метками классов и предсказанными метками модели (формула (1), где p – истинное распределение, а q – предсказанное распределение). Цель состоит в том, чтобы минимизировать эту функцию потерь, чтобы модель могла правильно классифицировать данные.

$$H(p, q) = -\sum_i p_i \log(q_i) \quad (1)$$

Экспериментально было выяснено, что самым эффективным оптимизатором для задачи классификации жестов является Adam (Adaptive Moment Estimation). Он применяется для обновления весов в нейронных сетях, чтобы минимизировать функцию потерь. Adam вычисляет скользящее среднее моментов градиентов первого и второго порядков, что позволяет ему адаптироваться к различным условиям оптимизации [14]. Он продемонстрировал высокую скорость сходимости и хорошее качество обучения (рис. 21).

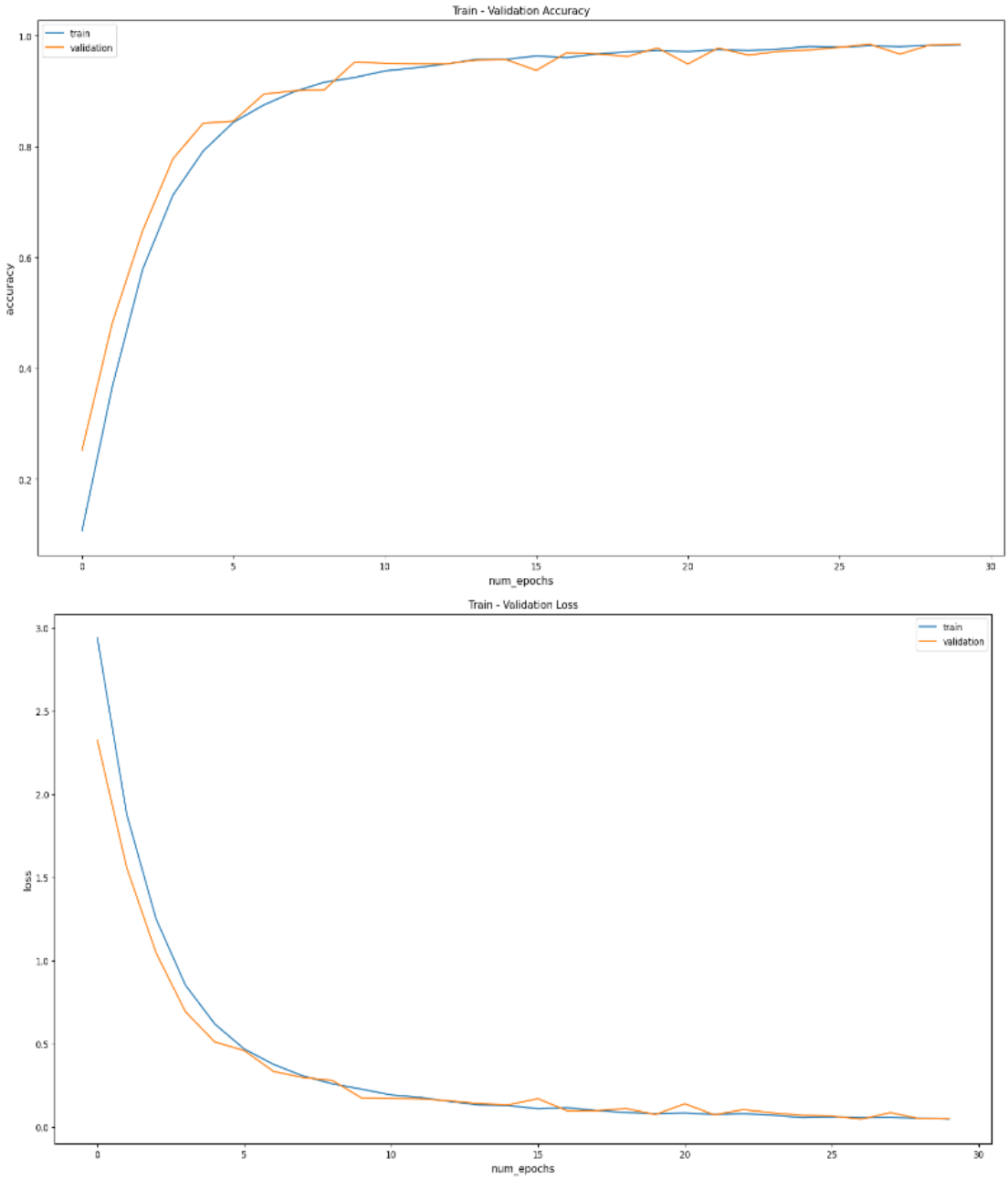

Рис. 21 - Пример графиков сходимости обучения и функции потерь

После компиляции модели важно настроить гиперпараметры. Гиперпараметры – это параметры модели, которые не могут быть обучены в процессе обучения и должны быть настроены заранее. Для настройки гиперпараметров в нейронных сетях существует несколько подходов. Один из наиболее популярных подходов - это использование метода случайного поиска (random search) или метода оптимизации гиперпараметров (hyperparameter optimization). Эти методы позволяют исследовать пространство гиперпараметров и найти оптимальные

значения для каждого из них [15]. Можно настроить следующие параметры обучения:

- Количество эпох – это количество раз, которое набор данных будет проходить через модель во время обучения. Подбирается путем многократного тестирования и выбора наилучшего результата.
- Размер пакета – это количество примеров, которые будут обработаны за одну итерацию обучения. Оптимальный размер пакета зависит от доступной памяти и размера набора данных.

Для настройки параметров обучения был выбран подход случайного поиска. Он заключается в случайной выборке значений из заданных диапазонов значений, а затем обучении модели с каждым набором гиперпараметров. Результаты обучения модели оцениваются на основе выбранных метрик качества, и выбирается набор гиперпараметров с наилучшими результатами.

При исследовании пространство гиперпараметров, для каждой модели на каждом этапе экспериментов были найдены оптимальные значения количества эпох и размера пакета. Они приведены в таблицах 1 – 3.

1. Классификация жестов цифр:

Таблица 1

Гиперпараметры моделей классификации жестов цифр

Модель	**RSL**		**ASL**	
	Количество эпох	**Размер пакета**	**Количество эпох**	**Размер пакета**
MLP	15	10	10	10
1D CNN	70	32	15	32
LSTM	100	32	150	32
2D CNN	30	32	15	32

2. Классификация жестов букв:

Таблица 2

Гиперпараметры моделей классификации жестов букв

Модель	RSL		ASL	
	Количество эпох	**Размер пакета**	**Количество эпох**	**Размер пакета**
MLP	15	32	15	1000
1D CNN	30	32	10	1000
LSTM	150	100	30	1000
2D CNN	10	32	5	1000

3. Классификация жестов цифр и букв:

Таблица 3

Гиперпараметры моделей классификации жестов цифр и букв

Модель	RSL		ASL	
	Количество эпох	**Размер пакета**	**Количество эпох**	**Размер пакета**
MLP	20	32	15	1000
1D CNN	30	32	15	1000
LSTM	100	32	45	1000
2D CNN	10	32	10	1500

Существует множество метрик качества, которые используются для оценки различных задач машинного обучения. Для задач классификации часто применяются метрики точности, полноты и F-меры. Точность (precision) – это доля объектов, которые классификатор отнес к положительному классу и при этом действительно являются положительными. Полнота (recall) – это доля объектов положительного класса, которые были правильно определены классификатором. F-мера (F-measure) – это гармоническое среднее между точностью и полнотой. Она позволяет получить более сбалансированную характеристику модели, чем две предыдущие метрики. F-мера достигает максимума при точности и полноте, равными единице, и близка к нулю, если один из аргументов близок к нулю [16]. Полноту, точность и F-меру можно вычислить по формулам 2 – 4 исходя из матрицы ошибок (рис. 22).

$$\text{Полнота} = \frac{TP}{TP+FP} \quad (2)$$

$$\text{Тосность} = \frac{TP}{(TP+FN)} \quad (3)$$

$$F - \text{мера} = \frac{TP}{TP+\frac{FP+FN}{2}} \quad (4)$$

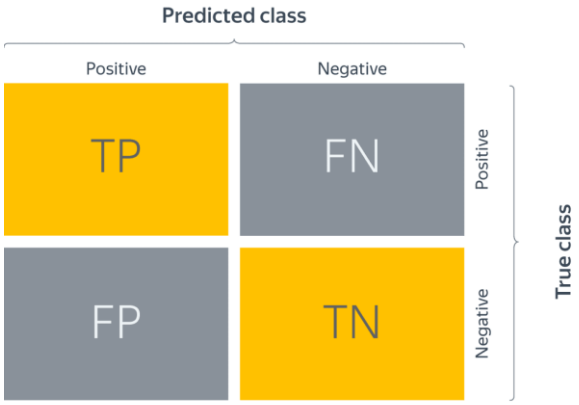

Рис. 22 - Матрица ошибок

Проверка качества обучения каждой нейронной сети проводилась на соответствующей тестовой выборке на каждом этапе экспериментов. Данная оценка модели показывает, насколько хорошо сеть обобщает данные и способна предсказывать результаты для новых данных. Кроме того, были произведены испытания на реальных видеоданных. Для каждого жестового языка заранее были записаны и размечены по кадрам соответствующие видео. Такое опробование является критически важным для обеспечения высокой производительности каждой модели в реальных условиях эксплуатации.

Ниже на таблицах 4 – 9 представлены вычисленные метрики качества для каждой модели нейронной сети на каждом этапе исследования:

1. Классификация жестов цифр:

Таблица 4

Оценка качества моделей на тестовых данных цифр

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.94	0.94	0.94	0.99	0.99	0.99
1D CNN	0.93	0.92	0.92	0.99	0.99	0.99
LSTM	0.61	0.60	0.59	0.96	0.96	0.96
2D CNN	0.99	0.99	0.99	1.00	1.00	1.00

Таблица 5

Оценка классификации жестов цифр на видео

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.81	0.79	0.78	0.39	0.34	0.30
1D CNN	0.78	0.78	0.77	0.10	0.18	0.11
LSTM	0.46	0.50	0.43	0.14	0.15	0.12
2D CNN	0.60	0.60	0.59	0.16	0.24	0.17

Анализ:

- Все модели нейронных сетей лучше обучаются на цифрах американского жестового языка;
- Лучшее качество классификации на корпусах данных в формате ключевых точек рук;
- Модели MLP и 1D CNN лучше всего показали себя в классификации цифр на видео в реальном времени;
- Модели, обучившиеся на цифрах русского жестового языка, лучше классифицируются на видео в реальном времени. Скорей всего это объясняется несколькими вещами. Во-первых, что в выбранном корпусе данных не было предложено видео с показом жестов цифр для тестирования, поэтому оно снималось самостоятельно. Поскольку американский жестовый язык мало знаком, то, возможно, на видео жесты показывались не совсем чётко и правильно. Во-вторых, в наборе данных ASL, на котором обучали модели, на всех фото одна и та же рука и на одном и том же чёрном фоне, то есть

малое разнообразие данных. В наборе же RSL разнообразия больше, за счёт использования рук разных форм и расположение их на цветных фонах с разным уровнем освещённости.

2. Классификация жестов букв:

Таблица 6

Оценка качества моделей на тестовых данных букв

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.99	0.99	0.99	1.00	1.00	1.00
1D CNN	0.96	0.95	0.94	1.00	1.00	1.00
LSTM	0.89	0.88	0.88	0.95	0.95	0.95
2D CNN	1.00	1.00	1.00	1.00	1.00	1.00

Таблица 7

Оценка классификации жестов букв на видео

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.12	0.24	0.11	0.33	0.41	0.30
1D CNN	0.43	0.52	0.44	0.32	0.38	0.30
LSTM	0.00	0.07	0.01	0.17	0.21	0.15
2D CNN	0.04	0.09	0.05	0.00	0.04	0.00

Анализ:

- На реальном видео лучше показали себя корпуса данных с ключевыми точками рук;
- Модели 1D CNN и MLP лучше всего показали себя в классификации букв на видео в реальном времени;
- Хуже всего качество модели LSTM и 2D CNN, с помощью них сложно классифицировать как жесты русских, так и английских букв на реальном видео. Причина может быть в малом количестве данных и плохо подобранных параметров сети;
- Несмотря на то, что корпус данных для американского алфавита имеет большее количество данных, модель 2D CNN очень плохо

классифицирует их на реальном видео. Возможно, дело в однообразности рук и фонов, используемых в наборе данных.

3. Классификация жестов цифр и букв:

Таблица 8

Оценка качества моделей на тестовых данных цифр и букв

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.96	0.96	0.96	1.00	1.00	1.00
1D CNN	0.94	0.92	0.92	1.00	0.99	0.99
LSTM	0.85	0.85	0.85	0.88	0.88	0.88
2D CNN	0.99	0.99	0.99	1.00	1.00	1.00

Таблица 9

Оценка классификации жестов цифр и букв на видео

Модель	RSL			ASL		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.27	0.26	0.25	0.22	0.16	0.15
1D CNN	0.24	0.30	0.23	0.12	0.09	0.09
LSTM	0.12	0.14	0.11	0.08	0.07	0.06
2D CNN	0.17	0.19	0.16	0.03	0.06	0.03

Анализ:

- Все модели нейронных сетей довольно плохо классифицируют и цифры, и буквы на реальных видео. Для видео, где показывались русские жесты, это может объяснятся тем, что было малое количество данных для обучения и взято мало ракурсов фотографий, а для американских жестов слишком однообразные данные и также мало ракурсов;
- Все модели классифицируют лучше цифры и буквы русского жестового языка, чем американского. Это может объяснятся тем, что корпуса данных для цифр и букв американского жестового языка довольно однообразны в форме рук, чем наборы данных русского;

– Хуже всего на видео показали себя модели LSTM и 2D CNN. Причиной может быть недостаточное количество данных и неправильный выбор параметров обучения.

5.3. Аугментация данных

Аугментация данных - это процесс создания новых образцов данных путем преобразования существующих. Это может помочь улучшить качество моделей нейронных сетей за счет увеличения количества доступных данных для обучения [14].

Поскольку корпус данных для русского жестового языка мал, то было решено провести эксперименты с применением процесса аугментации и переобучить каждую модель на новых данных. Использование такого инструмента может улучшить устойчивость моделей к изменениям входных данных и улучшить их общую производительность.

Для решения задачи классификации жестов по ключевым точкам рук были взяты методы девиации по координатам x и y в пределах 5% и сдвиг точек на угол в 2, 5 и 10 градусов по формулам 5 – 6, где random – случайное число от 0 до 1, а α – угол сдвига в градусах.

$$\begin{aligned} x &= x + 0.05 * random - \frac{0.05}{2} \\ y &= y + 0.05 * random - \frac{0.05}{2} \end{aligned} \tag{5}$$

$$\begin{aligned} x &= x * \cos\alpha - y * \sin\alpha \\ y &= x * \sin\alpha + y * \cos\alpha \end{aligned} \tag{6}$$

А для классификации по изображениям – случайным образом изменения яркости и контраста, и переворачивание по горизонтали (слева направо) и вертикали (сверху вниз).

Таким образом, после применения аугментации данных каждому экземпляру данных соответствует 5 образцов, где один оригинальный, а четыре аугментированных. Следовательно, наборы данных для цифр и букв русского жестового языка увеличились в 4 раза.

Так как входных данных для сетей стало больше, то необходимо перенастроить гиперпараметры для каждой модели русского жестового языка, чтобы не произошло переобучение или недообучение. При исследовании пространство гиперпараметров, для каждой модели на каждом этапе экспериментов были найдены новые оптимальные значения количества эпох и размера пакета. Они приведены в таблицах 10 – 12.

1. Классификация жестов цифр:

Таблица 10

Гиперпараметры моделей классификации русских жестов цифр

Модель	RSL	
	Количество эпох	**Размер пакета**
MLP	50	32
1D CNN	120	128
LSTM	150	64
2D CNN	15	64

2. Классификация жестов букв:

Таблица 11

Гиперпараметры моделей классификации русских жестов букв

Модель	RSL	
	Количество эпох	**Размер пакета**
MLP	30	64
1D CNN	50	128
LSTM	170	128
2D CNN	15	64

3. Классификация жестов цифр и букв:

Таблица 12

Гиперпараметры моделей классификации русских жестов цифр и букв

Модель	RSL	
	Количество эпох	**Размер пакета**
MLP	50	248
1D CNN	70	248
LSTM	170	512
2D CNN	30	128

Метрики качества также были пересчитаны для каждой модели русского жестового языка, и они приведены в таблицах 13 – 18.

1. Классификация жестов цифр:

Таблица 13

Оценка качества моделей на тестовых данных цифр

Модель	До аугментации			После аугментации		
	Точность	Полнота	F-мера	Точность	Полнота	F-мера
MLP	0.94	0.94	0.94	0.94	0.94	0.94
1D CNN	0.93	0.92	0.92	0.94	0.93	0.93
LSTM	0.61	0.60	0.59	0.83	0.80	0.79
2D CNN	0.99	0.99	0.99	0.99	0.99	0.99

Таблица 14

Оценка классификации русских жестов цифр на видео

Модель	До аугментации			После аугментации		
	Точность	Полнота	F-мера	Точность	Полнота	F-мера
MLP	0.81	0.79	0.78	0.84	0.86	0.84
1D CNN	0.78	0.78	0.77	0.80	0.82	0.80
LSTM	0.46	0.50	0.43	0.77	0.72	0.70
2D CNN	0.60	0.60	0.59	0.65	0.65	0.63

2. Классификация жестов букв:

Таблица 15

Оценка качества моделей на тестовых данных букв

Модель	До аугментации			После аугментации		
	Точность	Полнота	F-мера	Точность	Полнота	F-мера
MLP	0.99	0.99	0.99	1.00	0.99	1.00
1D CNN	0.96	0.95	0.94	0.98	0.97	0.97
LSTM	0.89	0.88	0.88	0.94	0.94	0.94
2D CNN	1.00	1.00	1.00	1.00	1.00	1.00

Таблица 16

Оценка классификации русских жестов букв на видео

Модель	До аугментации			После аугментации		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.12	0.24	0.11	0.41	0.50	0.42
1D CNN	0.43	0.52	0.44	0.52	0.61	0.53
LSTM	0.00	0.07	0.01	0.09	0.19	0.09
2D CNN	0.04	0.09	0.05	0.17	0.24	0.17

3. Классификация жестов цифр и букв:

Таблица 17

Оценка качества моделей на тестовых данных цифр и букв

Модель	До аугментации			После аугментации		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.96	0.96	0.96	0.97	0.97	0.97
1D CNN	0.94	0.92	0.92	0.95	0.94	0.94
LSTM	0.85	0.85	0.85	0.93	0.92	0.92
2D CNN	0.99	0.99	0.99	0.99	0.99	0.99

Таблица 18

Оценка классификации русских жестов цифр и букв на видео

Модель	До аугментации			После аугментации		
	Точность	**Полнота**	**F-мера**	**Точность**	**Полнота**	**F-мера**
MLP	0.27	0.26	0.25	0.29	0.36	0.31
1D CNN	0.24	0.30	0.23	0.22	0.28	0.23
LSTM	0.12	0.14	0.11	0.15	0.14	0.13
2D CNN	0.17	0.19	0.16	0.26	0.20	0.17

В результате проведения экспериментов с применением аугментации данных для задачи классификации цифр и букв русского жестового языка было продемонстрировано улучшение качества моделей. Повышение качества можно отследить на основе сравнения результатов работы нейронных сетей до и после применения аугментации данных.

Сравнительные таблицы показали, что применение аугментации данных позволило существенно улучшить производительность моделей, снизить переобучение и повысить устойчивость к изменениям входных данных. Таким образом, применение аугментации данных для данной задачи является эффективным инструментом для улучшения качества моделей.

Заключение

Алгоритм распознавания жестов рук на видео является важной темой в области компьютерного зрения. В работе были рассмотрены различные методы и подходы к решению этой задачи, включая методы машинного обучения и методы обработки изображений.

В итоге, в соответствии с поставленной задачей был разработан алгоритм с использованием четырёх созданных моделей нейронных сетей: MLP, 1D CNN, LSTM и 2D CNN, который способен определять жесты некоторых цифр и букв русского жестового языка. Была проведена оценка качества алгоритма на основе анализа его производительности, посчитанных метрик и на основе сравнения применения данного алгоритма с использованием данных американского жестового языка.

Кроме того, в ходе выпускной квалификационной работы был самостоятельно создан корпус данных русского жестового языка. Это позволило расширить возможности алгоритма распознавания жестов рук на видео и улучшить его производительность в условиях работы с русскоязычными пользователями. Применение аугментации данных также было важным этапом в создании этого набора, так как позволило увеличить количество доступных образцов данных и улучшить качество моделей машинного обучения.

Дальнейшее развитие алгоритма может включать улучшение его точности распознавания, создание более подходящих моделей классификации жестов, а также расширение его функциональности для работы с другими типами жестов и объектов. Таким образом, алгоритм распознавания жестов рук на видео имеет большой потенциал для применения в различных областях, связанных с компьютерным зрением, управлением компьютером и социальной интеграцией. В будущем, он может помочь снизить социальную изоляцию и улучшить качество жизни людей с нарушениями слуха и речи.

Список использованных источников

1. ВОЗ «Глухота и потеря слуха». [Электронный ресурс] – Режим доступа: https://www.who.int/ru/news-room/fact-sheets/detail/deafness-and-hearing-loss (дата обращения 08.05.2023);

2. Miah A.S.M., Hasan M.A.M., Shin, J., Okuyama, Y., Tomioka Multistage Spatial Attention-Based Neural Network for Hand Gesture Recognition // Computers 2023. Vol. 12, no. 13;

3. Abdallah M.S., Samaan, G.H., Wadie A.R., Makhmudov F., Cho Y. Light-Weight Deep Learning Techniques with Advanced Processing for Real-Time Hand Gesture Recognition // Sensors 2023. Vol. 23, no. 2;

4. Toro-Ossaba A., Jaramillo-Tigreros J., Tejada J.C., Peña A., López-González A., Castanho R.A. LSTM Recurrent Neural Network for Hand Gesture Recognition Using EMG Signals // Applied Sciences 2022. Vol. 12, no. 19;

5. Noreen I., Hamid M., Akram U., Malik S., Saleem M. Hand Pose Recognition Using Parallel Multi Stream CNN // Sensors 2021. Vol. 21, no. 24;

6. М. Г. Гриф, Р. Элаккия, А. Л. Приходько, М. А. Бакаев, Е. Раджалакшми Распознавание русского и индийского жестовых языков на основе машинного обучения // Системы анализа и обработки данных 2021. Том 83, № 3. – С. 53-74;

7. Kaggle. [Электронный ресурс] – Режим доступа: https://www.kaggle.com (дата обращения 09.05.2023);

8. Документация MediPipe. [Электронный ресурс] – Режим доступа: https://developers.google.com/mediapipe/solutions/vision/hand_landmarker (дата обращения 09.05.2023);

9. Сухачевская А.Г. Нейронные сети и их применение в решении задач АПК // Тенденции развития науки и образования 2021. С. 142;

10. H. Taud, J.F. Mas Multilayer Perceptron (MLP) // Geomatic Approaches for Modeling Land Change Scenarios 2018. Pp. 451 – 455;

11. Imene Mitiche, Alan Nesbitt, Stephen Conner, Philip Boreham, Gordon Morison 1D-CNN based real-time fault detection system for power asset diagnostics // The Institution of Engineering and Technology // Special Issue: Advanced Data-Analytics for Power System Operation, Control and Enhanced Situational Awareness 2020. Vol. 14. Issue 24;

12. Alex Sherstinsky Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network // Physica D: Nonlinear Phenomena 2020. Vol. 404;

13. Jing Chang, Jin Sha An efficient implementation of 2D convolution in CNN // IEICE Electronics Express 2017. Vol. 14. No. 1;

14. Документация TensorFlow. [Электронный ресурс] – Режим доступа: https://www.tensorflow.org (дата обращения 14.05.2023);
15. Раднаев Ч. Б Оптимизация гиперпараметров сверточной нейронной сети // XV Международная научно-практическая конференция студентов аспирантов и молодых учёных «Молодёжь и современные информационные технологии» 2017. С. 81 – 82;
16. Губко П., Горчаков А., Буркина М. Метрики классификации и регрессии. [Электронный ресурс] // Яндекс // Яндекс.академия – Режим доступа: https://academy.yandex.ru/handbook/ml/article/metriki-klassifikacii-i-regressii (дата обращения 20.05.2023).