Санкт-Петербургский государственный университет

Кафедра технологии программирования

Доржиев Зоригто Жаргалович

Выпускная квалификационная работа магистра

Анализ аудиоданных и распознавание событий для систем безопасности

Направление 02.04.02

Технологии баз данных

Научный руководитель

кандидат технических наук

Гришкин В. М.

Санкт-Петербург

2020

Содержание

Введение 3

Постановка задачи 7

Обзор литературы 11

Глава 1. Подготовка данных 13

1.1. Наборы данных 13

1.2. Скачивание данных 15

Глава 2. Характеристики аудио 18

2.1. Характеристики, связанные со временем 18

2.2. Спектральные характеристики 20

2.3. Извлечение характеристик 25

Глава 3. Классификация 30

3.1. Предыдущие результаты 30

3.2. Используемые методы 31

3.3. Data Augmentation 32

3.4. Batch Normalization 34

3.5. Dropout 35

3.6. Transfer Learning 35

3.7. Архитектура сети 36

Глава 4. Тестирование и результаты 40

Заключение 45

Список литературы 47

Введение

Современные методы обработки информации сделали большой шаг вперёд в различных задачах обработки и анализа данных. В этом постоянно возрастающем объёме цифровой информации особую роль играет аудио, так как около 20% информации человек получает через слух [1]. Существует огромное количество различных стриминговых платформ и сервисов, которые предоставляют доступ к мультимедийному контенту в разных формах.

Всё это привело к тому, что появляется необходимость в разработке различных методов и систем для автоматического анализа такого контента. Новые техники и подходы помогают решать большой спектр задач: распознавание речи, поиск информации на основе аудиофайлов, мультимодальный анализ, классификация аудиофайлов, сегментация, распознавание событий для систем безопасности и автоматизации процессов и т.д.

Ранние работы на тему извлечения музыкальной информации использовали символьные представления или нотации, такие как MIDI файлы [2]. С символьными представлениями было довольно легко работать, так как они не требуют высоких производительных мощностей. Это привело к разработке инструментов для синтаксического анализа таких представлений. Монофонические и полифонические транскрипции помогали работать с аудио, используя анализ символьных представлений. Однако распространяемая цифровым путём музыка в основном имеют форму неструктурированных аудиофайлов.

Различные исследования показали, что слушатели обращают внимание не на отдельные ноты, а на другие аспекты звука, которые пропадают из поля зрения автоматических систем, делающих упор на музыкальную теорию [3]. Ни одна из

систем, использующих монофонические и полифонические транскрипции, не была достаточно успешной для работы с сигналами реального мира.

Развивающаяся междисциплинарная наука извлечения информации из аудио объединила различные области: информатику, машинное обучение, обработку сигналов, психологию, психоакустику. Дисциплина имеет множество практических приложений для категоризации, манипуляции и даже для генерации новой информации.

Методы, которые основываются на семантическом сходстве экземпляров, используются для создания рекомендательных систем [4]. Раннее такие системы основывались только на метаданных – информации об исполнителе, жанре, годе выпуска и т.д. Другой подход использовал информацию о прослушиваниях других пользователей и делал предложения, основываясь на соответствующих коллекциях. Современные системы позволяют взглянуть на внутреннюю структуру сигнала и анализируют непосредственно характеристики аудио.

Извлечённые характеристики помогают решить задачу сепарации трека без доступа к изначальной студийной версии. Соответствующие программы, могут распознать и разделить трек на отдельные инструменты. Таким образом создаются караоке версии музыкальных композиций, однако качество не всегда бывает идеальным, потому что диапазон частот вокала находится внутри диапазонов некоторых других инструментов.

Часть приложений делает упор на автоматической транскрипции музыки, то есть процесса приведения аудиозаписи к символьной нотации. Результатом работы таких программ могут быть данные о ритме, мелодии, гармоническая информация и конечные MIDI файлы. Эта задача усложняется при увеличении количества инструментов в миксе и высокой полифонии в случаях, когда независимые мелодии накладываются друг на друга.

Помимо этого, проводится большое количество исследований в области автоматической генерации аудио. Также, как и с автоматическим созданием изображений разработанные алгоритмы имеют ограниченный успех с точки зрения человеческого восприятия и оценки результата. Например, в 2019 году в день рождения немецкого композитора Иоганна Себастьяна Баха компания Google выпустила интерактивный Doodle. Пользователи могли выбрать ноты, которые использовались для составления композиции в стиле известного композитора. Модель обучалась на наборе из 306 композиций Иоганна Себастьяна Баха, составляя закономерности при помощи алгоритмов машинного обучения [5].

Область машинного обучения, связанная с использованием глубоких нейросетей, проникает во все сферы. На примере набора данных MNIST (Modified National Institute of Standards and Technology database) для классификации рукописных цифр появляются аналогичные наборы аудиоданных для анализа произнесённых цифр и пола говорящего [6, 7].

Данная работа фокусируется на решении задачи классификации аудио событий. В отличии от классификации музыки, совокупность классов событий не ограничивается жанрами, инструментами, а принимает во внимание и аудиозаписи с другим содержанием. Множество классов, которое рассматривается в работе, варьируется от звуков природы и животных до звуков городской среды. Некоторые из таких классов имеют большой интерес в области безопасности. Например, своевременное распознавание звуков стрельбы или шума разбившегося стекла может помочь соответствующим службам вовремя отреагировать на экстренную ситуацию.

В современном мире системы безопасности можно встретить повсюду. Некоторые из таких систем имеют возможность записывать не только видео, но

и аудио. Аудиосигнал после обработки может нести информацию, которая будет помогать работе “умного дома”.

Распознаванию звуковых событий уделяется немалый интерес в последние годы. Появляются приложения, которые решают задачи мониторинга в области здравоохранения, анализа городских звуков и даже отслеживания популяции птиц [8, 9].

Постановка задачи

Возрастающее количество и доступность аудиоданных привели к появлению автоматических систем, которые их анализируют. Целями квалификационной работы являются

- Разработка и проверка работоспособности алгоритма для классификации аудиоинформации. Созданный алгоритм должен автоматически присваивать любой аудиозаписи один или несколько заранее заданных классов.

Для достижения этих целей были поставлены следующие задачи:

1. Поиск репрезентативного набора данных.
2. Извлечение характеристик аудиофайлов.
3. Применение и сравнение алгоритмов классификации.
4. Перенос знаний модели, обученной на большом наборе данных.

Самый первый шаг в работе очень важен, потому что от правильного выбора набора данных напрямую зависит результат работы алгоритмов. В рамках работы необходимо рассмотреть наборы данных с хорошей выборкой классов городских звуков.

Аудио может хранится во многих форматах, которые имеют различное предназначение. Данные форматы представления звуковых данных хранят информацию о частоте и амплитуде звука. Они отличаются друг от друга степенью сжатия и направленностью на профессиональное или бюджетное звуковоспроизводящее оборудование.

Существует три основных формата аудиофайлов:

- Форматы без сжатия, WAV, AIFF, RAW.
- Форматы со сжатием без потерь, FLAC, M4A.

- Форматы со сжатием звука с потерями, MP3, AAC.

Несмотря на то, что MP3 является очень популярным форматом и используется повсеместно в файлообменных сетях, принцип сжатия значительно снижает точность частей звукового потока, которые считаются трудноразличимыми для человеческого уха. Психоакустическая модель позволяет корректировать степень сжатия в зависимости от пределов восприятия звука. Считается, что нижний предел восприятия равен 16 Гц, а верхний предел – 20000 Гц. От уровня звукового давления, который измеряется в децибелах (дБ), и частоты зависит абсолютный порог слышимости [10].

Звукозаписывающие устройства способны захватывать аудио за пределами порога слышимости человека, что позволяет использовать дополнительную информацию в исследовании.

В данной работе используются аудиофайлы в формате WAV, который был разработан Microsoft и IBM в 1991 году. WAV файлы являются одним из примеров контейнерного формата RIFF (Resource Interchange File Format). Аудиофайлы в формате WAV хранят аудио со сжатием и без сжатия, хотя наиболее распространённой является оцифровка без сжатия с использованием импульсно-кодовой модуляции (pulse code modulation, PCM).

Линейная импульсно-кодовая модуляция (linear pulse code modulation, LPCM) широко используется в цифровой звукозаписи. Например, WAV, MP3, FLAC и другие форматы используют импульсно-кодовую модуляцию во время преобразования аналогового сигнала с компакт-диска в цифровой. Частота дискретизации таких файлов – 44100 Гц с 16 битами на семпл. В профессиональной звукозаписи формат WAV с LPCM используется для достижения высокого качества аудио.

С форматом WAV работает большинство устройств, также файлы в данном формате довольно легко редактировать и обрабатывать с помощью специального программного обеспечения.

Исходные данные исследования – это набор одноканальных аудиофайлов, приведённых к формату WAV. Каждому аудиофайлу ставится в соответствие вектор $\boldsymbol{y}$, который хранит информацию о классах звуковых событий, присутствующих в записи.

Звуковое событие – это класс или метка, которая отвечает на вопрос, что происходит на аудиозаписи. Звуковые события помогают понять и описать структуру аудио, соответствующей сцены (улица, небольшая комната). Например, оживлённая улица может содержать следующие звуковые события: звук проезжающих автомобилей и гудков, шаги и речь людей. Каждое звуковое событие можно описать с различной степенью информативности. Класс “музыка” может содержать большое количество подклассов, описывающих жанр, конкретный инструмент или даже часть мира, где можно услышать определённое звуковое событие. Класс “речь” может дополнительно описывать говорящего с помощью подклассов “мужской голос”, “женский голос” и т. д.

Изначальный набор аудиосигналов используется для извлечения характеристик. Характеристики аудиофайла могут быть связаны со временем, частотой или темпом. После объединения характеристик для каждого экземпляра получается вектор $\boldsymbol{x}$, который содержит численную информацию и может использоваться как входной вектор для классификаторов и нейросетей.

Задачу классификации можно постепенно усложнять и переходить от бинарной классификации к мультиклассификации. Для решения первой задачи необходимо разделить набор данных на два класса, например, звуковые события, которые связаны с человеческой речью и другие. Далее в модель добавляются

новые классы, и каждый аудиофайл может содержать несколько меток событий, что существенно усложняет задачу.

В данной работе должны быть рассмотрены популярные алгоритмы классификации. Получившиеся модели сравниваются между собой. Веса этих моделей можно использовать, как начальные веса для других наборов данных.

Конечная модель должна классифицировать поступающий аудиосигнал в режиме реального времени. Результатом работы программы является вектор предсказанных моделью классов. В выборе лучшего метода должна будет учитываться точность и время работы алгоритма.

Обзор литературы

1. *Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. Emre Çakir, Giambattista Parascandolo, Toni Heittola, Heikki Huttunen, Tuomas Virtanen. 2017.*

В данной статье авторы решают задачу классификации аудио событий (Sound Event Detection, SED) при помощи свёрточных нейронных сетей (Convolutional Neural Networks, CNN). Нейронные сети помогают извлекать характеристики более высокого уровня, которые являются инвариантными к локальным временным или спектральным изменениям. Объединив свёрточные нейронные сети с рекуррентными сетями (Recurrent Neural Networks, RNN), авторы предлагают метод CRNN, который значительно улучшает производительность для четырёх наборов повседневных звуковых событий. Для оценки классификатора использовались F-мера и равный уровень ошибок. Несмотря на уменьшение уровня ошибок до 11% для набора данных CHIME-HOME, авторы указывают на ограничения, связанные с объёмом доступных данных. Также они считают, что использование метода Transfer Learning может улучшить результат работы алгоритмов. Эта идея рассматривает в данной квалификационной работе.

2. *pyAudioAnalysis: An Open-Source Python Library for Audio Signal Analysis. Theodoros Giannakopoulos. 2015.*

Данная статья описывает открытую библиотеку для анализа аудиосигналов, написанную на языке программирования Python. Библиотека позволяет извлекать аудио характеристики, а также предоставляет высокоуровневые и простые в использовании оболочки для различных задач, таких как классификация, регрессия или сегментация.

Помимо этого, в состав библиотеки входят инструменты для визуализации аудио данных. Автор использует данную работу для задачи определения эмоции в речи, автоматического создания превью для трека.

Характеристики извлечённые при помощи pyAudioAnalysis используются в квалификационной работе, а классификационные модели на основе метода опорных векторов и k-ближайших соседей, полученные автором библиотеки, сравниваются с новыми обученными нейросетями.

3. *Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification. Justin Salamon, Juan Pablo Bello. 2016*

В этой статье авторы описывают архитектуру свёрточной сети и алгоритмы, необходимые для препроцессинга данных. Предложенная архитектура состоит из 3 свёрточных слоёв, за которыми следуют два полносвязных слоя. Авторы используют методы увеличения данных, чтобы достигнуть лучшего результата на небольшом наборе Urban-Sound8K. Были применены методы сдвига записи по времени, добавления шума, которые также используются и в этой квалификационной работе. Хотя средняя точность работы алгоритма была невысокой – 74%. Для отдельных классов авторам удалось достичь 90%.

Глава 1. Подготовка данных.

1.1 Наборы данных

Существует много различных наборов данных для анализа музыки, например: Free Music Archive, Million Song Dataset [13, 14]. Для анализа речи можно использовать Free Spoken Digit Dataset [7], Voxceleb [15], The Spoken Wikipedia Corpora [16]. Наборы данных отличаются размером (от 10 мегабайт до нескольких терабайт), наличием метаинформации и типом данных, в которых этот набор представлен (некоторые наборы состоят только из заранее вычисленных характеристик).

Однако вышеперечисленные датасеты весьма однородны, поэтому выбор упал на масштабный набор данных AudioSet [17], предоставленный Google. AudioSet состоит из более 2 миллионов вручную размеченных десятисекундных звуковых клипов, загруженных на видеохостинг YouTube. Каждому отрывку ставятся в соответствие классы аудиособытий (музыка, речь, звук автомобиля и т.п.) Количество уникальных классов – 527. Онтология покрывает широкий спектр звуков человека и животных, музыкальных инструментов и жанров, а также повседневных звуков окружающей среды.

Набор данных AudioSet разделён на три непересекающихся выборки: два набора для сбалансированного обучения и оценки результата и набор для несбалансированного обучения. Первые два набора данных содержат около 20 тысяч отрывков, на каждый класс приходится как минимум 59 экземпляров. Несбалансированный набор представляет собой оставшуюся часть датасета.

На рисунке 1 изображено распределение классов по количеству экземпляров в AudioSet. Самые крупные классы связаны с музыкой и речью, тогда как количество экземпляров для некоторых специфических звуков не превышает 200.

Рис.1 Количество экземпляров по классам в AudioSet

AudioSet предоставляет данные для загрузки в двух форматах:

- CSV-файл, описывающий для каждого экземпляра: ID видео на YouTube; начальное и конечное время отрезка; классы, к которым принадлежит данный экземпляр.
- 128-размерные характеристики аудио, извлечённые с частотой дискретизации 1 Гц. Для их извлечения была использована VGG-подобная модель, которая работала с YouTube-8M (набором из 6 миллионов размеченных отрывков для анализа видео).

Для того, чтобы самостоятельно извлечь характеристики и работать непосредственно с аудиофайлами, было решено использовать первый вариант.

Модель, обученная на наборе данных AudioSet, далее используется для работы с набором данных городских звуков Urban Sound Classification [18]. Этот набор содержит 8732 звуковых фрагмента в формате WAV, которые принадлежат 10 классам.

1.2. Скачивание данных

В отличии от Urban Sound Classification, который предоставляет доступ к самим аудиофайлам, в AudioSet входят три CSV-файла для сбалансированного и несбалансированного обучения и проверки результата.

Внутренняя структура файлов выглядит следующим образом:

```
['---1_cCGK4M', ' 0.000', ' 10.000', '/m/01g50p', '/m/07rwm0c']
```

Каждый элемент набора представлен в виде строки файла. Первый элемент строки это ID, которое используется для загрузки видео при помощи YouTube API. Второй и третий элемент – это время начала и конца отрывка. Далее следует один или несколько идентификаторов классов, к которым принадлежит рассматриваемый сегмент.

В данном примере первые 10 секунд видео с ID “---1_cCGK4M” содержат звуковое событие, принадлежащее классам “/m/01g50p” (Railroad car, train wagon, звук поезда) и “/m/07rwm0c” (Clickety-clack, быстрые и ритмичные щелчки машин, звук стальных колёс).

Видеохостинг YouTube хранит аудио- и видеопотоки отдельно друг от друга. Эти потоки отличаются друг от друга разрешением, количеством кадров в секунду, используемым кодеком или битрейтом. Для получения ссылки на

аудиопоток с битрейтом 160 кбит/с (максимальный битрейт для большинства экземпляров в наборе) была использована библиотека pytube, которая предлагает простые в использовании инструменты для загрузки видео с YouTube для Python [19].

Идентификатор видео передаётся pytube, который возвращает список аудиопотоков. Этот список сортируется по битрейту, а ссылка на поток с максимальным битрейтом используется для скачивания аудио отрывка.

Для того, чтобы не хранить аудиопотоки целиком и использовать только размеченные сегменты, необходимо предварительно отредактировать каждый аудиопоток.

Временные ссылки на нужный поток максимального качества, полученные с помощью pytube, передаются дальше набору свободных библиотек FFmpeg, который используется для записи и конвертации аудио- и видеофайлов в различных форматах [20]. FFmpeg является популярным решением для кодирования, декодирования, фильтрации и воспроизведения файлов.

Информация о времени начала и конца рассматриваемого сегмента, полученная из CSV-файла, используется FFmpeg для сегментации десятисекундных отрывков. Таким образом, FFmpeg получает на вход ссылку на аудиопоток и две временные метки. По умолчанию аудиокодек использует импульсно-кодовую модуляцию с 16 битами на семпл (pcm_s16le). Далее каждый отрывок конвертируется в одноканальный и скачивается в формате WAV с частотой дискретизации 16000 Гц.

Во время скачивания аудиофайлов возникло несколько ошибок:

- Часть видео была удалена с видеохостинга или недоступна по какой-либо другой причине.

- Время окончания сегмента превышает длину видео. Для того, чтобы все сегменты были одинаковой длительности, а вектора характеристик совпадали по размерности, такие случаи игнорируются.
- YouTube блокирует большое количество запросов к серверам и может заблокировать доступ к ним. Десятисекундные паузы между запросами решают эту проблему.

В итоге было загружено около 30000 фрагментов общим объёмом более 10 Гб. Эти фрагменты были отсортированы по классам в соответствии с файлом онтологии - ontology.json. Файл хранит:

- Машинный идентификатор для каждого класса, например: “/m/0dgw9r”.
- Название класса. “Speech”, “music”, “guitar”, “gunshot” и т.д.
- Описание класса.
- Ссылку на цитирование из описания.
- Ссылки на примеры класса.
- Идентификаторы подклассов. Например “Human voice”, “female voice” и “male voice” – это три разных класса.
- Ограничения для классов, классификация которых может быть затруднительна по какой-либо причине.

Глава 2. Характеристики аудио

2.1 Характеристики, связанные со временем

Существует множество различных характеристик для аудио. Они могут быть связаны со временем, частотой или темпом. В основном, временные характеристики извлекаются напрямую из аудиосигнала. В данной работе были использованы следующие характеристики.

Energy – сумма квадратов значений сигнала, нормализованная по длине фрейма. Пусть $x_i(n)$, n = 1, …, W_L – это последовательность аудио семплов фрейма i, где W_L – это длина фрейма. Тогда значение энергии сигнала E, можно вычислить по формуле:

$$E(i) = \frac{1}{W_L} \sum_{n=1}^{W_L} |x_i(n)|^2 \qquad (2.1)$$

Уравнение (2.1) описывает мощность сигнала. Эта характеристика имеет тенденцию больших скачков в последовательных окнах сигнала для определённых классов. Например, речь может содержать большое количество пауз и слабых фонем.

Zero crossing rate, ZCR – количество перемен знака аудиосигнала на промежутке, то есть количество раз, когда сигнал меняет своё значение с положительного на отрицательное или наоборот. ZCR определяется следующим уравнением:

$$\mathrm{Z}(i) = \frac{1}{2W_L} \sum_{n=1}^{W_L} |sgn[x_i(n)] - sqn[x_i(n-1)]|, \qquad (2.2)$$

Где sgn – это кусочно-постоянная функция, определяемая следующим уравнением:

$$sgn[x_i(n)] = \begin{cases} 1, & x_i(n) \geq 0, \\ -1, & x_i(n) < 0. \end{cases} \quad (2.3)$$

ZCR можно интерпретировать, как меру зашумленности, и эта характеристика обычно принимает большие значения в случае шумных сигналов. Помимо этого, ZCR может с некоторой точностью выразить спектральные характеристики сигнала. Эти свойства и невысокая вычислительная сложность привели к тому, что ZCR применяется во многих задачах от обнаружения речи до классификации музыкальных жанров.

Entropy of Energy – краткосрочная энтропия энергии, мера резких изменений уровня энергии сигнала. Чтобы вычислить энтропию энергии, каждый фрейм делится на K частей зафиксированной длительности. Затем для каждого под-фрейма j, рассчитывается энергия $E_{subFrame_j}$ и делится на полную энергию $E_{shortFrame_i}$. Получившаяся последовательность значений энергии для под-фреймов e_j, j = 1, … , K выражается соотношением:

$$e_j = \frac{E_{subFrame_j}}{E_{shortFrame_i}}, \quad (2.4)$$

где

$$E_{shortFrame_i} = \sum_{k=1}^{K} E_{subFrame_k} \quad (2.5)$$

На последнем шаге энтропия H(i) последовательности e_j вычисляется согласно уравнению (2.6):

$$H(i) = -\sum_{j=1}^{K} e_j \cdot log_2(e_j) \quad (2.6)$$

Получившееся значение уменьшается при резких изменениях в энергии. Поэтому данная характеристика может быть использована для обнаружения момента начала выстрела, взрыва или других подобных звуков с крупными и быстрыми изменениями значений энергии.

Дополнительно, исследования показывают, что минимальное значение энтропии меняется в зависимости от музыкального жанра. Например, минимум энтропии для классической музыки больше, чем для электронной музыки. Это может быть объяснено тем, что электронная музыка содержит большее количество резких переходов и скачков энергии.

2.2 Спектральные характеристики

Следующие характеристики связаны с частотой и называются спектральными. Для того, чтобы вычислить спектральные характеристики необходимы дискретные преобразования Фурье. Они широко используются в анализе аудиосигналов и обеспечивают удобное представление частотного распределения звука, то есть его спектра.

Пусть $X_i(k)$, k = 1, … , W_{fL} – это коэффициенты дискретного преобразования Фурье для фрейма i.

Spectral Centroid – центр тяжести спектра. Значения этой характеристики можно определить по формуле:

$$C_i = \frac{\sum_{k=1}^{W_{fL}} kX_i(k)}{\sum_{k=1}^{W_{fL}} X_i(k)} \quad (2.7)$$

Spectral Spread – это второй центральный момент спектра, значения которого находятся с помощью отклонений спектра от центра тяжести спектра согласно уравнению (2.8):

$$S_i = \sqrt{\frac{\sum_{k=1}^{W_{fL}} (k - C_i)^2 X_i(k)}{\sum_{k=1}^{W_{fL}} X_i(k)}} \quad (2.8)$$

Получившаяся характеристика измеряет, как спектр распределён рядом с его центроидой. Малые значения описывают спектр, сконцентрированный у своего центра тяжести.

Аналогично энтропии энергии можно рассчитать спектральную энтропию, *Spectral Entropy*. Спектр фрейма делится на L меньших диапазонов. Энергия E_f диапазона f такого, что f = 0, … , L – 1, нормализуется по полной энергии спектра:

$$n_f = \frac{E_f}{\sum_{f=0}^{L-1} E_f} \quad (2.9)$$

Энтропия нормализованной энергии спектра вычисляется по формуле:

$$H = -\sum_{f=0}^{L-1} n_f \cdot log_2(n_f) \quad (2.10)$$

Spectral Flux – спектральный поток, измеряющий изменения спектра в двух последовательных фреймах. Spectral Flux измеряется как возведённая в квадрат разность величин спектра двух последовательных краткосрочных окон:

$$Fl_{(i,i-1)} = \sum_{k=1}^{W_{fL}} (EN_i(k) - EN_{i-1}(k))^2, \quad (2.11)$$

где $EN_i(k)$ – это нормализованные коэффициенты дискретного преобразования Фурье фрейма i:

$$EN_i(k) = \frac{X_i(k)}{\sum_{l=1}^{W_{fL}} X_i(l)} \quad (2.12)$$

Спектральный поток принимает более высокие значения для классов, связанных с речью. Это может быть связано с быстрым чередованием фонем и шумной природой звука в других классах.

Spectral Rolloff, спектральный спад – частота, ниже которой находится определённый процент значений спектра. Если коэффициент дискретного преобразования Фурье с индексом m соответствует спектральному спаду фрейма i, тогда он удовлетворяет следующему уравнению:

$$\sum_{k=1}^{m} X_i(k) = C \sum_{k=1}^{Wf_L} X_i(k), \quad (2.13)$$

где C – это пользовательский параметр, отвечающий за процент. Спектральный спад обычно нормализуется и принимает значения от 0 до 1. Эта характеристика описывает форму спектра и используется для разделения типов звуковых событий.

Mel-Frequency Cepstrum Coefficients (MFCCs) – мел-частотные кепстральные коэффициенты, которые вычисляются согласно графику

зависимости высоты звука от частоты колебаний. Расчёт коэффициентов проводится в несколько шагов.

1. Вычисляются дискретные преобразования Фурье.
2. Получившийся спектр подаётся на вход L фильтрам, основанным на мел-шкале. Мел-шкала вводится для того, чтобы учитывать эффект деформации частот в соответствии с психоакустическими наблюдениями. Слуховая система человека проще различает разницу между более низкими частотами. Таким образом, высота звука, которая воспринимается человеком, не линейно зависит от частоты. Ниже представлена одна из формул, которая пытается выразить эту зависимость:

$$f_w = 1127.01048 \cdot \log(\frac{f}{700} + 1) \qquad (2.14)$$

График на рис. 2 приблизительно описывает полученную зависимость.

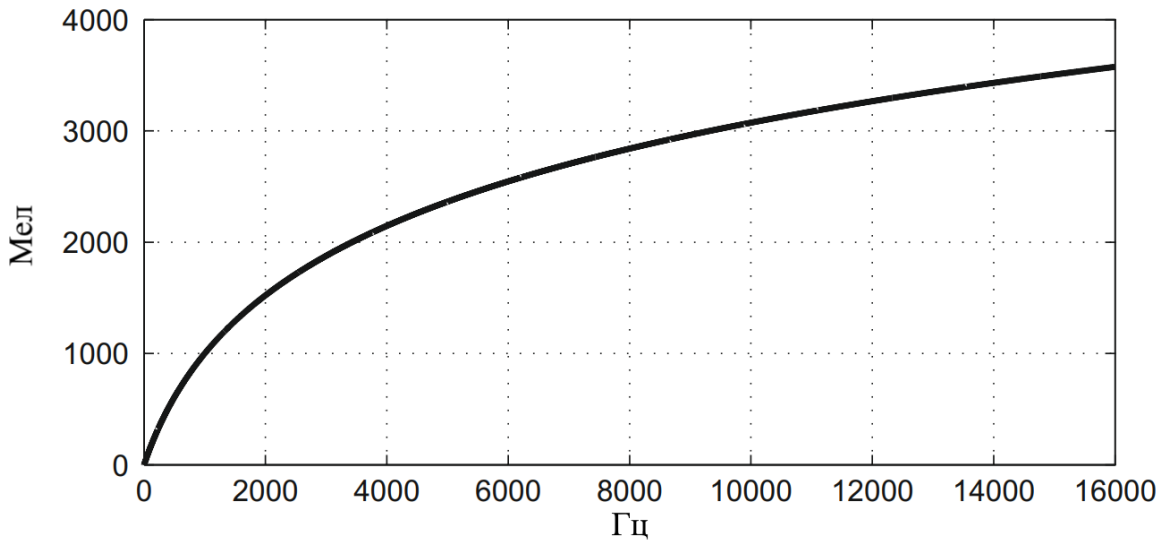

Рис. 2 Высота звука в зависимости от его частоты.

3. Если $\hat{O}_k$, k = 1, … , L, – это мощность на выходе фильтра k, тогда мел-частотные кепстральные коэффициенты удовлетворяют выражению:

$$c_m = \sum_{k=1}^{L} (\log \hat{O}_k) \cos\left[m\left(k - \frac{1}{2}\right)\frac{\pi}{L}\right], m = 1, \dots, L. \quad (2.15)$$

Мел-частотные кепстральные коэффициенты широко применяются в обнаружении речи, кластеризации говорящих на записи, классификации музыкальных жанров и во многих других задачах анализа аудио. Стоит отметить, что в зависимости от задачи могут применяться разные наборы коэффициентов. Например, во многих приложениях, которые работают с музыкой, выбираются первые 13 мел-частотных кепстральных коэффициентов, потому что они несут достаточное количество информации для многих классификационных задач.

Chroma Vector – это представление энергии спектра, состоящее из 12 элементов. Коэффициенты дискретного преобразования Фурье группируются по 12 интервалам. Каждый интервал по европейской музыкальной традиции соответствует полутону. Для интервалов можно рассчитать средние логарифмические значения коэффициентов Фурье:

$$v_k = \sum_{n \in S_k} \frac{X_i(n)}{N_k}, \qquad k \in 0, \dots, 11, \quad (2.16)$$

где S_k – это подмножество частот, которые соответствуют коэффициентам дискретного преобразования Фурье, а N_k – это мощность этого подмножества.

Векторы v_k можно объединить в матрицу V, которая называется хромограмма, по аналогии со спектрограммой. Визуально хромограммы для речи более зашумлены по сравнению с хромограммами для музыки.

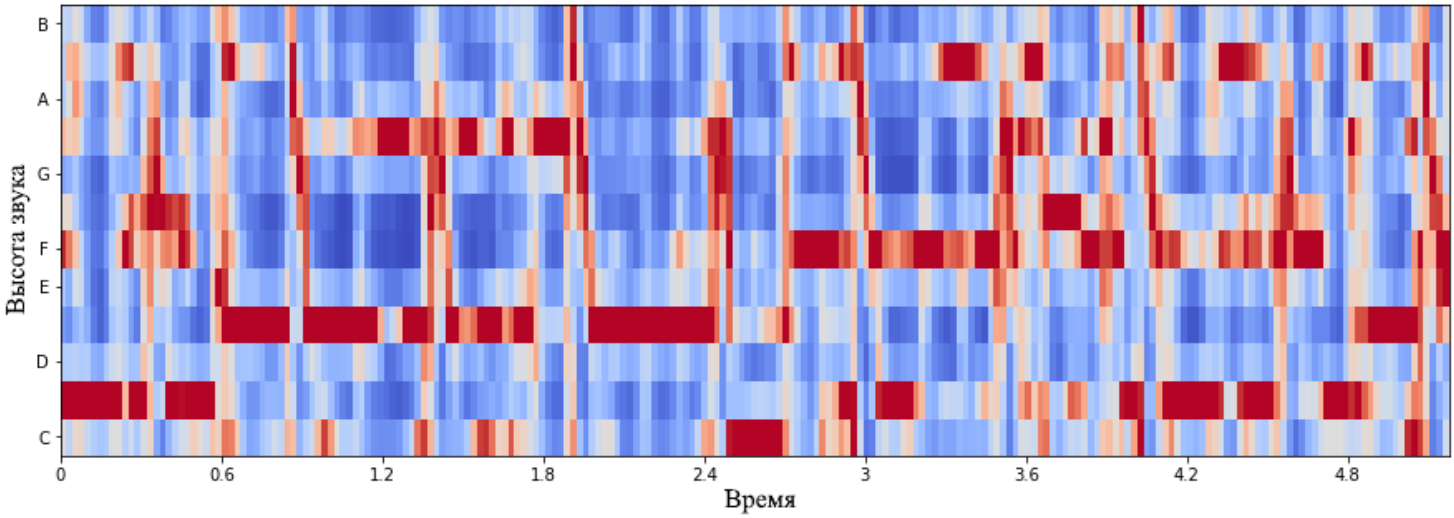

Рис. 3 Хромограмма для музыкальной композиции.

На рисунке 3 изображена хромограмма музыкального трека, на которой видно, что сигнал придерживается определённых музыкальных интервалов.

2.3 Извлечение характеристик

Аудиосигнал сначала разделяется на короткие сегменты или фреймы, затем для каждого фрейма извлекаются искомые характеристики аудио. В результате получаются последовательности краткосрочных векторов характеристик. Размер таких сегментов может меняться, однако широко распространены окна длительностью от 20 до 100 миллисекунд.

Для извлечения характеристик была использована библиотека pyAudioAnalysis [12]. Полученный результат можно представить в виде графика, изображённого на Рисунке 4. На графике изображены две характеристики: ZCR и энергия.

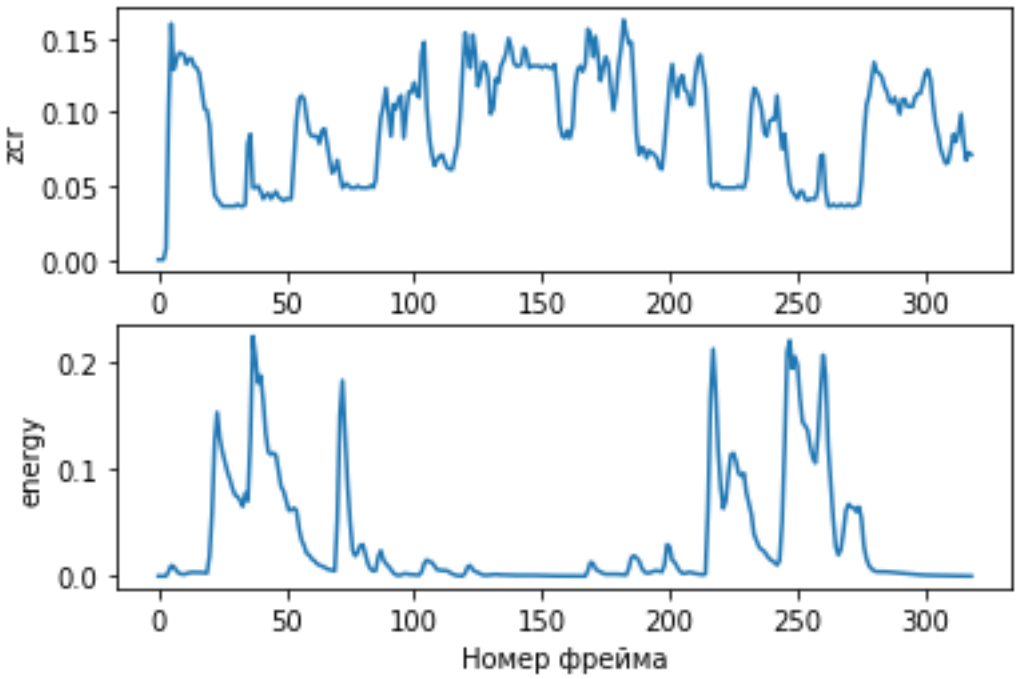

Рис. 4 Значения Zero crossing rate и Energy в зависимости от фрейма

Библиотека pyAudioAnalysis позволяет также выводить спектрограммы аудиосигналов. На рисунке 5 изображён пример такого представления аудиофайла.

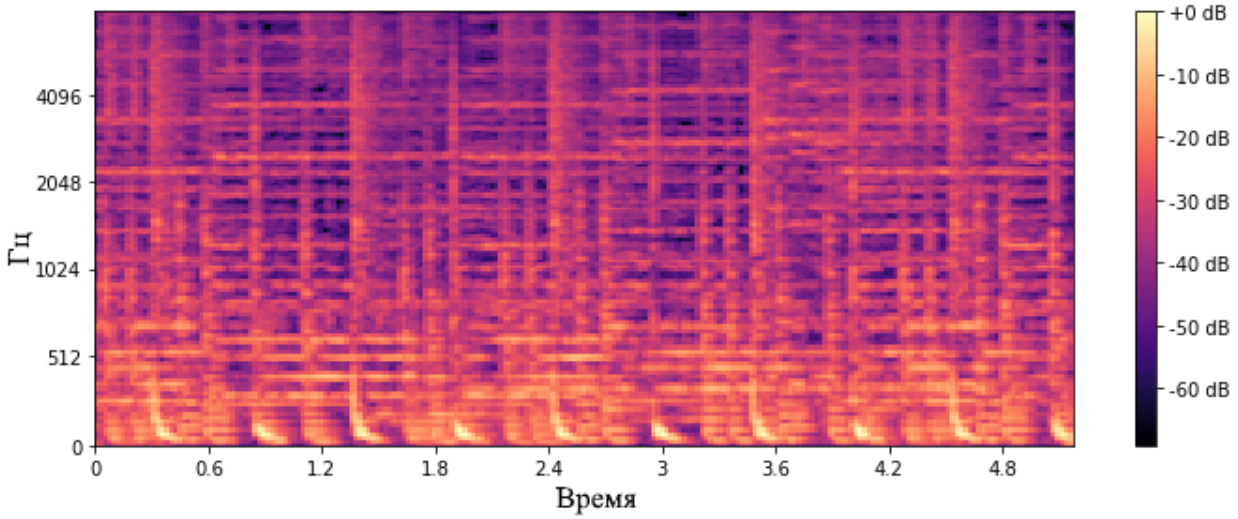

Рис 5. Спектрограмма аудиосигнала.

В pyAudioAnalysis характеристики рассчитываются на небольших отрезках с наложением друг на друга (overlapping) или без (non-overlapping). В первом случае длина шага меньше длины окна, а во втором случае – они равны.

Ещё один подход, который применяется в анализе аудио, – это обработка последовательностей характеристик на основе сегментов среднего размера. Эти сегменты также могут накладываться друг на друга. Для каждого сегмента рассчитываются краткосрочные характеристики на маленьких окнах, после чего находятся средние значения для каждой характеристики. Далее для представления этих сегментов среднего размера употребляются полученные наборы статистик.

Стандартные значения длины среднего сегмента варьируются от 1 до 10 секунд. Для более длинных аудиозаписей можно вычислять долгосрочные характеристики, то есть находить среднее по сегментам среднего размера.

Для набора данных AudioSet в данной работе были использованы краткосрочные характеристики, потому что длительность аудиозаписей не превышает 10 секунд. Последовательность характеристик извлекалась с длительностью фрейма – 50 миллисекунд, и длиной шага – 25 миллисекунд. Таким образом, процент наложения сегментов друг на друга равен 50%.

Функция, извлекающая характеристики в pyAudioAnalysis, возвращает 68 характеристик. После их извлечения каждый аудиофайл может быть представлен в виде матрицы. Для десятисекундных отрывков размер матрицы составляет 68x400.

Скорость извлечения характеристик невысока, например, для обработки 6500 экземпляров набора данных потребовалось около 2 часов. Однако эта скорость всё ещё позволяет обрабатывать аудиосигналы в режиме реального времени.

Полученный массив с характеристиками для всех аудиозаписей далее хранится в структуре данных DataFrame из библиотеки pandas. Первый столбец таблицы соответствует имени файла, а во втором столбце хранятся численные значения самих характеристик.

В третьем столбце хранятся векторы *y* правильных классов, к которым относится аудиозаписи. Для построения векторов используется файл онтологии.

Таблицу с данными и извлечёнными характеристиками можно легко сохранить для последующей работы. Пример таблицы для бинарной классификации музыки и всех остальных классов можно увидеть на рисунке 6.

	filename	features	music
0	18739.wav	[[0.08760951188986232, 0.09386733416770963, 0....	True
1	18740.wav	[[0.11639549436795996, 0.10262828535669587, 0....	False
2	18741.wav	[[0.2065081351689612, 0.2202753441802253, 0.24...	False
3	18742.wav	[[0.02753441802252816, 0.023779724655819776, 0...	True
4	18743.wav	[[0.04005006257822278, 0.08385481852315395, 0....	True

Рис. 6. Первые пять элементов в таблице для бинарной классификации.

В ходе исследования было обнаружено, что в исходной выборке для многих звуковых событий не указаны родительские классы. Таким образом, дочерние классы, которые отвечают за музыкальные инструменты, не относятся к родительскому классу – музыка.

Для корректного описания векторами всех звуковых событий, необходимо чтобы они содержали информацию о каждом классе, к которому принадлежат аудиозаписи. Каждый класс рекурсивно проверяется на наличие подклассов, после чего создаётся иерархия.

Объединение классов в более крупные группы позволяет упростить задачу на первом шаге. Данный подход помогает свести задачу мультиклассификации к бинарной классификации, например, научиться различать звуковые события, связанные с речью и музыкой.

Глава 3. Классификация.

3.1 Предыдущие результаты

Существует много классификационных алгоритмов, которые с разной степенью успешности использовались в предыдущих исследованиях. В рассмотренной статье для классификации использовались два различных метода kNN и SVM [12].

Метод kNN (k-nearest neighbors algorithm) – это простой классификатор, который работает на основе поиска k-ближайших соседей. Рассматриваемому объекту присваивается класс, к которому относится большинство из k ближайших объектов.

Во время работы этого метода могут возникать сложности, связанные с выбранной метрикой. В случае звуковых событий, распределения характеристик отличаются друг от друга, поэтому их необходимо предварительно отнормировать.

Метод опорных векторов (SVM, support vector machine) – это ещё один линейный классификатор. Принцип его работы основан на построении гиперплоскости, которая разделяет пространство характеристик.

Эти алгоритмы довольно популярны и применяются в самых разных задачах. Реализацию методов предоставляет библиотека машинного обучения scikit-learn [21]. Размеченные аудиофайлы были отсортированы по разным папкам. Для обучения использовалось по 15000 экземпляров для классов Speech и Music.

Полученные результаты бинарной классификации предоставлены в Таблице 1.

Метод	Точность
kNN	93.1%
SVM	94.6%

Таблица 1. Результаты классификации

Результаты показывают, что для бинарной классификации метод опорных векторов превосходит по точности метод k-ближайших соседей.

3.2 Используемые методы

С увеличением количества информации и популяризации методов машинного обучения одними из наиболее точных алгоритмов стали нейросети. Их используют для анализа текста, изображений, нахождения дефектов, создания новых данных и т.д.

Со временем количество параметров, глубина нейросетей, количество скрытых слоёв только растёт. Одним из наиболее важных шагов было внедрение VGG-подобных моделей, которые будут описаны далее.

Для классификации и обучения модели невероятно важно то, насколько хорошо представлен набор данных. Например, в AudioSet могут наблюдаться большие различия внутри класса, в то время как разные классы могут иметь схожие характеристики. То есть появляется проблема высокой внутриклассовой вариации. Это означает, что характеристики объектов, принадлежащие одному и тому же классу, могут значительно отличаться друг от друга.

Например, класс “музыка” содержит и классические произведения, и электронные ремиксы, и отдельные звуки инструментов. Для таких звуковых событий разница в значениях частотных и временных характеристик может привести к ошибочной мисклассификации.

С другой стороны, набор данных AudioSet содержит экземпляры с высоким межклассовым сходством. Для некоторых классов, связанных с речью, требуется определить пол говорящего или даже его возраст.

Для того, чтобы справится с такой задачей во время обучения глубокой нейросети необходимо использовать нетривиальные методы. В частности, в данной работе были применены три метода для обучения небольшой свёрточной нейронной сети, которые одновременно повышали эффективность алгоритма и служили инструментом для регуляризации. Затем был произведён перенос весов для работы с меньшим набором данных.

3.3 Data Augmentation

Одним из недостатков использования методов машинного обучения является необходимость работы с расширенными наборами данных для лучшего обобщения результатов. Работая с небольшими наборами данных, модель способна быстро переобучиться, то есть достигнуть состояния, в котором с каждой новой эпохой точность для тренировочного набора растёт, а для тестового падает.

Переобученная модель будет давать ошибочные результаты, если все экземпляры тестового набора были получены в одинаковом окружении. С целью не допустить это, можно применить метод увеличения количества данных (Data augmentation).

Существуют различные способы трансформации аудиофайлов, среди которых добавление случайного шума, сдвиги по времени, изменение высоты звука и скорости записи.

Набор данных для сбалансированного обучения содержит только 22 тысячи экземпляров, что значительно меньше несбалансированного набора, в котором 2 миллиона фрагментов. Приведённые выше методы увеличения количества данных могут значительно повысить качество модели.

В этой работе было использовано наложение шума на аудиозаписи и изменение высоты звука. Добавление шума производится с помощью библиотеки NumPy [22], которая генерирует последовательность случайных значений. Элементы последовательности умножаются на параметр факторизации шума, который меняет силу, с которой случайный шум влияет на исходную аудиозапись. Для того, чтобы изменить высоту звука, была использована функция pitch_shift из библиотеки librosa [23]. Результат представлен на рисунке 7.

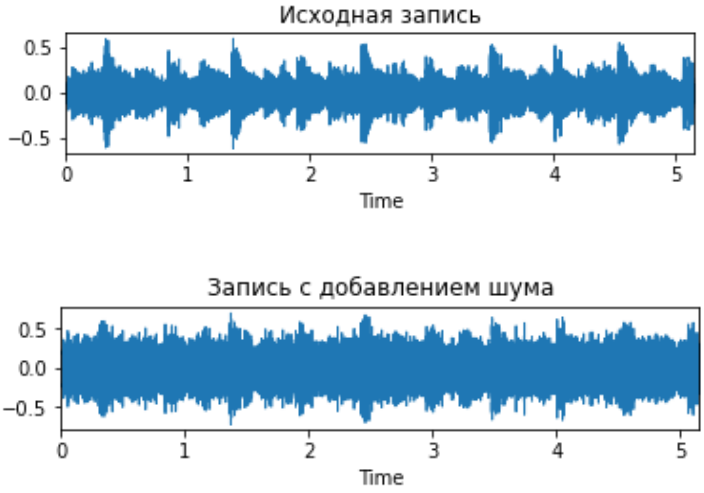

Рис. 7. Сравнение волновой формы аудиозаписи до и после добавления шума.

Этот алгоритм был применён к каждой аудиозаписи исходного набора. Таким образом, удалось увеличить количество данных для обучения в два раза.

3.4 Batch Normalization

С целью избежать ошибок, которые возникают при недостаточном количестве памяти, набор данных делится на небольшие пакеты или партии (batches). Метод пакетной нормализации (Batch Normalization) описывает процесс нормализации каждого пакета данных, суть которого заключается в приведении данных к нормальному распределению для каждого слоя.

Каждая партия данных нормализуется таким образом, чтобы среднее значение, математическое ожидание выборки, было равно нулю, а среднеквадратическое отклонение – единице ($\mu = 0, \sigma^2 = 1$).

Пакетная нормализация была введена как метод регуляризации, чтобы уменьшить значение, которое называется внутренним ковариационным сдвигом [24]. Это значение описывает сдвиг распределений в каждом из слоёв сети. Мотивацией для использования пакетной нормализации является уменьшение такого сдвига, что позволяет нейросети сходится быстрее.

Внутренний ковариационный сдвиг практически не мешает сходимости нейросети, как показывают текущие исследования [25]. Несмотря на это пакетная нормализация позволяет нейросети быть менее чувствительной к гиперпараметрам, сходиться быстрее и обучаться с более высокими коэффициентами обучения (learning rate).

Предполагается, что эти эффекты возникают из-за сглаживания градиента, которое выражается в небольших изменениях величины потерь. Пакетная нормализация – это полезный метод во время обучения. Во время тестирования эффекты могут быть полностью поглощены весами сети. Поэтому это метод используется только для обучения свёрточной нейросети.

3.5 Dropout

Для того, чтобы различные части модели специализировались на разных аспектах анализа данных, существует много методов. Один из способов достижения этого является случайная деактивация определённой части узлов в нейросети во время обучения. Таким образом, на каждом шаге обучения используется новая подсеть, а на этапе тестирования все узлы заново активируются. В результате нейросеть достигает меньшей ошибки обобщения (generalization error) [26]. Во время проверки результатов важно сохранять средний коэффициент активации постоянным.

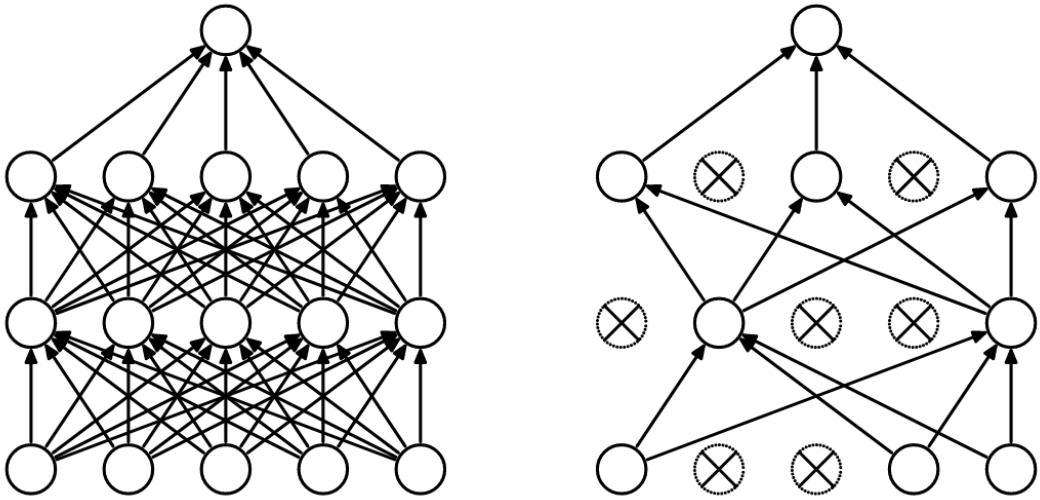

Рис. 8. Узлы исходной сети (слева) деактивируются, чтобы получить новую сеть (справа)

3.6 Transfer Learning

Как было отмечено ранее, огромную роль в обучении модели играют входные данные, от них напрямую зависят обобщающие способности сети. Небольшие наборы данных страдают от недостатка разнообразия, что мешает точности для обучения нейросетей с нуля. Модели попросту неспособны получить полезные высокоуровневые представления на таких небольших наборах данных.

Один из подходов, который позволяет преодолеть недостатки маленьких наборов, – это Transfer Learning. Если существует более крупный датасет с похожими данными, тогда появляется возможность использовать знания, полученные во время обучения на таком наборе данных.

Например, для анализа изображений существует набор данных ImageNet, который состоит из более 14 миллионов изображений и 20 тысяч категорий. Предварительно обученные модели на ImageNet включены в состав библиотеки PyTorch [27] и используются во многих работах. Модели способные классифицировать породы собак, различать ягоды от воздушных шаров, показывали отличные результаты и в других более прикладных задачах. Веса моделей, обученных на ImageNet, использовались как начальные веса для классификации дефектов на поверхности горячекатаной стали [28].

Модели, обученные на больших наборах данных, остаётся только настроить для более конкретной задачи. В данной работе модель изначально решала задачу классификации для более общего набора AudioSet, после чего веса сохранялись и переносились в другую модель, работающую с набором Urban Sound Classification. Таким образом, нейросеть лучше адаптируется к новым данным и использует обобщённые представления более высокого уровня.

3.7 Архитектура сети

В данном исследовании используются две архитектуры свёрточной нейронной сети. Первая модель должна быть простой, но тем не менее при обучении с нуля достигать высокой точности. Вторая нейросеть представляет собой VGG-подобную модель.

При построении простой модели учитывались следующие аспекты:

- С одной стороны, информация, необходимая для классификации резких звуков, сосредоточенна локально в небольшом временном промежутке.

- С другой стороны, для анализа речи или других классов с монотонными продолжительными звуковыми событиями требуется рассматривать более крупные фрагменты записи.

Поэтому размер свёрточных фильтров был значительно увеличен до 11 и 15, в то время как более широко используются размеры 3 и 5.

Дополнительно, для увеличения восприимчивого поля нейрона и одновременного уменьшения количества параметров и сложности используется метод выбора максимального элемента (max-pooling). Этот метод не должен помешать работе модели из-за наблюдаемого внутриклассового подобия.

Задачу классификации не решить, используя только низкоуровневые представления характеристик. Несмотря на то, что для резких звуков достаточно взглянуть на энергию сигнала, многие экземпляры имеют высокое межклассовое сходство, поэтому недостаточно использовать только один свёрточный шаг. Для того, чтобы получить более высокоуровневые представления из аудиозаписей, необходимо несколько шагов.

Данная модель не должна быть слишком сложной, потому что более глубокие сети будут рассмотрены далее. В итоге первый прототип модели состоит из двух свёрточных слоёв с max-pooling, за которыми следует плотная сеть прямой связи, используемая для классификации приведённых типов звуковых событий.

Также во время обучения используются упомянутые выше методы пакетной нормализации, случайной деактивации отдельных нейронов и увеличения количества данных. Параметры модели, такие как коэффициент деактивации и глубина каналов, настраивались в соответствии с начальными догадками, после чего выбирался наиболее эффективный набор параметров.

Вторая модель, которая была использована в расчётах, имеет VGG-подобную архитектуру. Группа таких нейросетей была разработана коллективом Visual Geometry Group в Оксфорде [29]. Они показывали лучшие результаты в задаче классификации изображений ImageNet. Их преимуществом над другой архитектурой AlexNet является отсутствие крупных фильтров, что делает модель намного глубже. Это привело к тому, что нейросеть создаёт более высокоуровневые абстракции, улучшая конечный результат.

Есть четыре основных типа VGG моделей: VGG 11, 13, 16 и 19. Каждая из этих моделей имеет соответствующее названию количество скрытых слоёв. Все модели состоят из множества свёрточных слоёв с max-pooling. В итоге информация разделяется на 512 каналов.

Библиотека PyTorch содержит инструменты для имплементации этих моделей и внедрения пакетной нормализации.

В данном исследовании сравниваются две архитектуры VGG 16 и VGG 11 для обнаружения зависимости точности от количества параметров нейросети. Благодаря своему размеру VGG 11 содержит меньшее количество параметров и, следовательно, требует меньшей вычислительной мощности и времени.

В исходных моделях последний слой заменяется в зависимости от задачи мульти- или бинарной классификации, чтобы соответствовать количеству категорий. Например, на выходе для бинарной классификации используются только два нейрона.

Однако на практике аудиозаписи могут принадлежать к нескольким классам звуковых событий и приходится решать задачу многозначной классификации. Метку для предсказанных классов можно представить в виде вектора. Например, вектор [1, 1, 0, 0 … 0] указывает на аудиозапись, которая

принадлежит к первым двум классам звуковых событий и не принадлежит к остальным.

При переносе весов для небольшого набора данных, существует две опции. В первом случае, веса замораживаются и обновляется только последний слой, в другом случае, вся модель переобучается полностью. Значения изменённого выходного слоя округляются для получения готового ответа, который можно сравнить с исходной меткой-вектором.

Во время обучения использовались две функции потерь: перекрёстная энтропия (cross entropy) и квадратичная функция потерь (MSE). Эти функции вычисляют разницу между исходным вектором и вектором, который был предсказан моделью. Перекрёстная энтропия была использована для бинарной классификации, в остальных случаях была применена квадратичная функция потерь.

Глава 4. Тестирование и результаты

Набор данных был разделён с помощью встроенной в библиотеку scikit-learn функции стратификации. 70 процентов данных было использовано для обучения, а 30 – для тестирования. Размер партии (batch size) равен 50. Следовательно, каждая эпоха состоит из 280 небольших партий. Тестирование проводилось на оставшихся 120 пакетах.

Функция потерь была оптимизирована с помощью алгоритма Adam, который использует продвинутые методы преодоления недостатков обычного стохастического градиентного спуска [30]. Для коэффициента скорости обучения было выбрано значение 10^{-4}. В качестве функции активации была использована ReLU, так как она обеспечивает лучшую сходимость и не страдает от угасания градиента.

На вход каждой сети подаётся двумерный массив характеристик размерностью 68x400. Результат на выходном слое отличается для задач бинарной классификации и многозначной классификации, где используются значения, округлённые до ближайшего целого.

Все вычисления были произведены на платформе Google Colab [31], которая предоставляет доступ к Tesla P4 GPU. Каждый подход был протестирован, как минимум на 40 эпохах. Если в какой-то момент времени функция потерь начинала расти на протяжении нескольких эпох обучение останавливалось. Далее подходы анализируются и сравниваются между собой.

Простой модели, обученной с нуля, потребовалось около 100 эпох для того, чтобы сойтись. Несмотря на то, что каждая эпоха занимала не более минуты, обучение простой модели заняло наибольшее количество времени и вычислительных мощностей из-за высокого количества эпох.

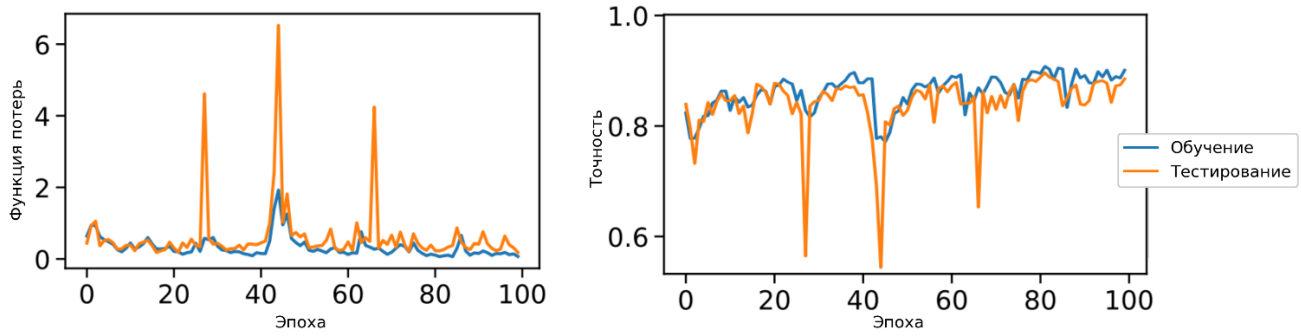

Рис. 9. График функции потерь и точности простой модели для бинарной классификации.

Множественные выбросы на графике функции потерь и точности указывают на то, что модель не закончила обучаться и достигла локального минимума во время оптимизации. После 30 эпох без резких выбросов обучение было остановлено, так как нейросеть показывала знаки сходимости. Дополнительные эпохи не привели к увеличению точности, однако модель не показывала признаков переобучения.

В итоге после 100 эпох простая модель достигла 92.1% точности для задачи бинарной классификации, для задачи мультиклассификации точность была ниже – 85.7%.

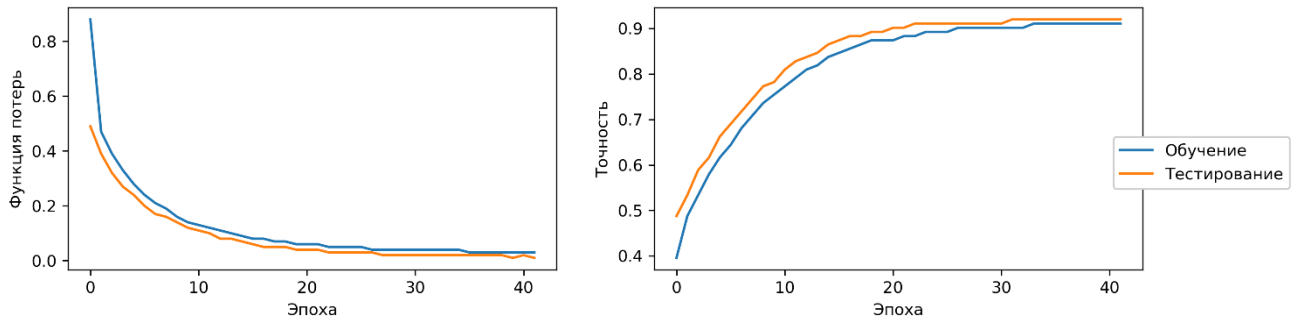

Рис. 10 График функции потерь и точности VGG 16 для мультиклассификации.

Кривая обучения для модели VGG 16 показывает более быструю сходимость. Так как количество параметров для VGG 16 намного выше, чем у предыдущей модели, каждая эпоха обучения занимала примерно 114 секунд, что почти в два раза превосходит время для простой модели. После 40 эпох нейросеть

достигла 98.8% точности в задаче бинарной классификации, для мультиклассификации – 92.6%.

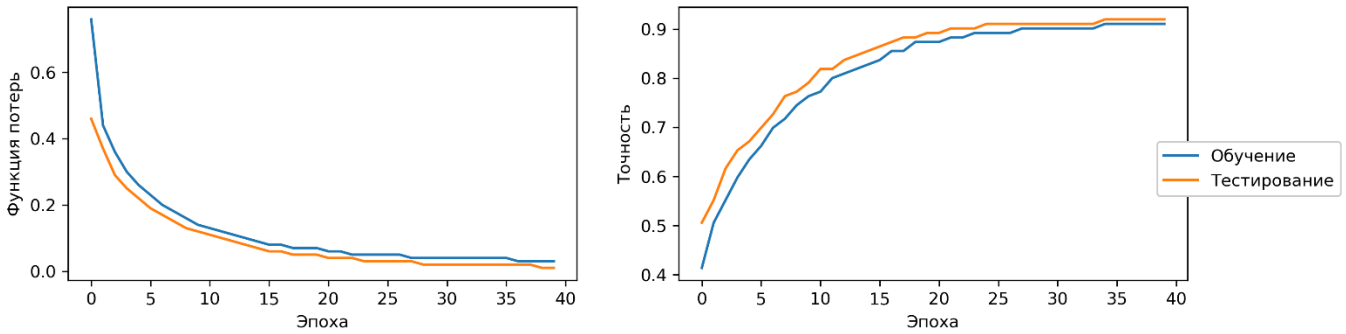

Рис 11. График функции потерь и точности VGG 11 для мультиклассификации. Кривые обучения VGG 11 и 16 похожи друг на друга. Тесты показали, что VGG 11 сходится быстрее.

Простейшей VGG модели потребовалось 75 секунд в среднем для каждой эпохи. VGG 11 сошлась за 30 эпох, что намного быстрее результата VGG 16.

После обучения моделей на наборе данных AudioSet их веса сохранялись и были использованы для работы с Urban Sound Classification. Во время тестирования первая модель страдала от переобучения после нескольких эпох. Что привело к значительной 15-процентной разнице в точности на этапах обучения и тестирования (см. Рис. 12).

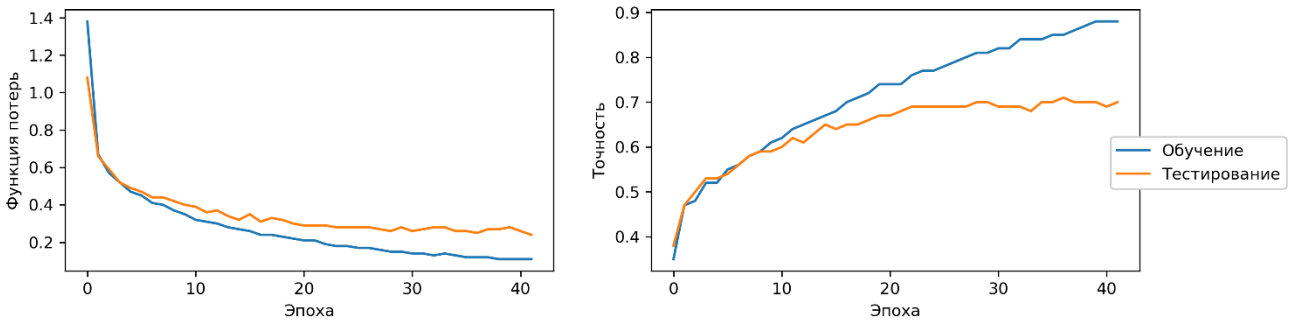

Рис. 12. При работе с набором Urban Sound Classification, модели переобучаются.

После 40 эпох и около 42 минут реального времени простая модель достигла 74% точности, что гораздо меньше результатов, полученных для набора данных AudioSet.

Предварительно обученные VGG модели, обучающиеся на расширенном наборе данных Urban Sound Classification, показали себя немного лучше, достигая 81-85%. Результаты VGG 11 и VGG 16 сравнивались между собой. Оказалось, что итоговые значения точности для VGG 16 не превосходят VGG 11. Более того, обучение модели с VGG 11 менее затратно по времени. Для сходимости VGG 16 потребовалось 34 минуты, а для VGG 11 – 28 минут.

Итоговые таблицы результатов выглядят следующим образом:

Модель	**Точность**	**Время работы**	**Количество эпох**
Простая нейросеть	92.1%	5404 сек.	96
VGG 11	97.9%	3866 сек.	37
VGG 16	98.8%	4558 сек.	40

Таблица 2. Результаты работы алгоритмов для бинарной классификации набора AudioSet.

Модель	Точность	Время работы	Количество эпох
AudioSet			
Простая нейросеть	85.7%	5943 сек.	108
VGG 11	92.0%	4452 сек.	40
VGG 16	92.6%	4865 сек.	40
Urban Sound Classification			
Простая нейросеть	73.9%	3757 сек.	42
VGG 11	82.5%	4396 сек.	40
VGG 16	85.3%	4820 сек.	40

Таблица 3. Результаты работы алгоритмов для мультиклассификации наборов AudioSet и Urban Sound Classification.

Заключение

В ходе работы удалось выполнить поставленные задачи. Были реализованы алгоритмы скачивания данных, извлечения характеристик. Помимо этого, были найдены решения для увеличения объёма выборки для небольших наборов данных.

Полученные модели, основанные на нейросетях, способны классифицировать широкий спектр звуковых событий. Они не уступают по показателям ранее разработанным алгоритмам на основе методов k-ближайших соседей и опорных векторов.

Модели, обученные на большом наборе AudioSet, неплохо показали себя для расширенного набора Urban Sound Classification. Предыдущие работы достигали 74% точности, предварительно обученным VGG моделям удалось улучшить этот результат, доведя точность до 85%.

Можно отметить, что увеличение количества скрытых слоёв в VGG-подобных моделях повышает точность лишь на небольшие значения. Возможно, стоит обратить внимание на более простые модели с меньшим количеством параметров, потому что для некоторых задач увеличение вычислительной сложности только повышает время обучения.

Классификация аудиозаписей происходит со скоростью, достаточной для работы в реальном времени. Следующими шагами в исследовании будут внедрение автоматического захвата аудио и одновременного анализа аудиопотока, что позволит классифицировать звуковые события, захваченные микрофоном на любом устройстве.

Реализованные классификаторы могут быть использованы в различных областях. Они могут помочь заранее оповестить службы об экстренной ситуации,

либо о состоянии пациента, который проходит мониторинг. Некоторые приложение даже включают в себя отслеживание миграции животных на основе звукового анализа.

Кроме того, подобные сети можно использовать для автоматической генерации новых аудиозаписей. Большой интерес составляет применение вариационных автоэнкодеров и генеративно-состязательных сетей в области музыки.

Вероятно, новейшие алгоритмы, основанные на остаточном обучении, и более тонкая настройка параметров смогут значительно повысить эффективность решения задач классификации и анализа аудиоданных.

Список литературы

1. Dale E. Audio-Visual Methods in Teaching, 3rd ed., Holt, Rinehart & Winston, New York, 1969, P. 108.
2. Tzanetakis G., Cook P. Audio Information Retrieval (AIR) Tools. 2002.
3. Martin K., Scheirer E., Vercoe B. Musical content analysis through models of audition. In Proc. ACM Multimedia Workshop on Content-Based Processing of Music, Bristol, UK, 1998.
4. Chen H., Chen A.L.P. A Music Recommendation System Based on Music and User Grouping. J Intell Inf Syst 24, 113–132 (2005).
5. День Иоганна Себастьяна Баха. https://www.google.com/doodles/celebrating-johann-sebastian-bach
6. LeCun Y., Cortes C., Burges CJ. MNIST handwritten digit database. // ATT Labs. Vol. 2. 2010.
7. Free Spoken Digit Dataset (FSDD). https://github.com/Jakobovski/free-spoken-digit-dataset
8. Goetze S., Schroder J., Gerlach S., Hollosi D., Appell J.-E., Wallhoff F. Acoustic monitoring and localization for social care // Journal of Computing Science and Engineering, vol. 6, no. 1, pp. 40–50.
9. 2012.Stowell D., Clayton D. Acoustic event detection for multiple overlapping similar sources // 2015 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA), 2015, pp. 1-5.
10. Jayant N., Johnston J., Safranek R. // "Signal Compression Based on Models of Human Perception". Proceedings of the IEEE. 81 (10): 1385–1422. (October 1993).
11. Çakir E., Parascandolo G., Heittola T., Huttunen H., Virtanen T. Convolutional Recurrent Neural Networks for Polyphonic Sound Event Detection. 2017.

12. Giannakopoulos T. pyAudioAnalysis: An Open-Source Python Library for Audio Signal Analysis. 2015.
13. Defferrard M., Benzi K., Vandergheynst P., Bresson X. FMA: A dataset for music analysis. 2017.
14. Bertin-Mahieux T., Ellis D., Whitman B., Lamere P. The Million Song Dataset // Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR). 2011.
15. Nagrani A., Chung J.S., Zisserman A. VoxCeleb: a large-scale speaker identification dataset // INTERSPEECH. 2017.
16. Baumann T., Köhn A., Hennig F. The Spoken Wikipedia Corpus collection: Harvesting, alignment and an application to hyperlistening // Language Resources and Evaluation. 2018.
17. Gemmeke J., Ellis D., Freedman D., Jansen A., Lawrence W., Moore R.C., Plakal M., Ritter M. Audio Set: An ontology and human-labeled dataset for audio events // Proc. IEEE ICASSP. 2017.
18. Salamon J., Jacoby C., Bello J. P. A Dataset and Taxonomy for Urban Sound Research // 22nd ACM International Conference on Multimedia. 2014.
19. pytube3 9.6.4 documentation. https://python-pytube.readthedocs.io/en/latest/
20. FFMpeg. https://ffmpeg.org/
21. scikit-learn: machine learning in Python. https://scikit-learn.org/stable/
22. NumPy. https://numpy.org/
23. Librosa. https://librosa.github.io/
24. Ioffe S., Szegedy C. Batch normalization: Accelerating deep network training by reducing internal covariate shift // Proceedings of Machine Learning Research, vol. 37. PMLR, 07–09 Jul 2015, pp. 448–456.
25. Santurkar S., Tsipras D., Ilyas A., Madry A. How does batch normalization help optimization? // Curran Associates, Inc., 2018, pp. 2483–2493.

26. G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors // CoRR, vol. abs/1207.0580, 2012.
27. PyTorch. https://pytorch.org/
28. K. Song and Y. Yan. A noise robust method based on completed local binary patterns for hot-rolled steel strip surface defects // Applied Surface Science, vol. 285, pp. 858 – 864, 2013.
29. K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition // CoRR, vol. abs/1409.1556, 2014.
30. D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” 2014.
31. Google Colaboratory. https://colab.research.google.com/