Санкт–Петербургский государственный университет

Кафедра компьютерного моделирования и многопроцессорных систем

Мирошниченко Александр Сергеевич

Выпускная квалификационная работа бакалавра

Разработка системы распознавания речевых команд при помощи методов машинного обучения

Направление 01.03.02

«Прикладная математика и информатика»

Научный руководитель:

кандидат физ.-мат. наук,

доцент

Козынченко В. А.

Рецензент:

доктор физ-мат. наук,

профессор

Котина Е. Д.

Санкт-Петербург

2021 г.

Содержание

Введение

В современном компьютеризированном мире огромное значение имеет взаимодействие человека с компьютером - ввод и вывод информации с устройства в понятной для человека форме. Один из способов внести информацию в компьютер - записать речь через микрофон, после чего можно обрабатывать данные в памяти, которые ее представляют. Обработка может быть совершенно разной, но особенно важно распознавать слова, которые произнёс человек и давать им представление в виде текста. То есть, давать такое представление речи, как если бы она была не произнесена голосом, а напечатана при помощи клавиатуры.

Решение такой задачи может быть использовано для различных целей. К примеру, можно переписываться с другим человеком по интернету текстовыми сообщениями и при этом вообще не прикасаться к клавиатуре, управлять различными компьютерными интерфейсами при помощи голосовых команд и т.д.

Данная проблема была актуальна со времён появления компьютеров и остаётся таковой по сей день. Особенно актуальна она стала в последнее время, когда появились качественные микрофоны, возросла мощность вычислительных устройств, увеличилось количество информации, которой обмениваются люди через интернет.

Изначально, для решения данной задачи применялись такие алгоритмы, как скрытые Марковские модели, методы динамического программирования, методы дискриминантного анализа, основанные на Байесовской дискриминации и другие. Но с появлением нейронных сетей и многочисленных экспериментов с их использованием выяснилось, что задачу распознавания речи можно решать и при помощи нейросетевого подхода. И хоть сами нейронные сети появились ещё в прошлом веке, их популярность возросла только в последнее время, в связи с ростом мощности компьютеров.

В данной работе ставится задача распознавания речи и предлагается её решение при помощи многослойного персептрона и свёрточной нейронной сети. В качестве решения подзадачи выделения характеристик речи предлагается алгоритм мел-частотных кепстральных коэффициентов, разработанный в 70-х годах прошлого века, учитывающий особенности слухового восприятия человеком.

Краткое содержание глав:

В главе 1 рассмотрены теоретические сведения о звуке, речевых признаках и общая схема алгоритма распознавания звука.

В главе 2 рассмотрен алгоритм распознавания речи, его программная реализация и датасет. Описаны подзадачи: предобработка звукового сигнала, выделение речевых признаков, распознавание речи.

В главе 3 приведены результаты вычислительных экспериментов.

Постановка задачи

Пусть $X = \{x_0, ..., x_{p-1}, ...\}$ - множество объектов речи. Оно состоит из векторов амплитуд $x_i = \{x_i^0, ..., x_i^{k_i}\},\ i = \overline{0, \inf}$. Здесь k_i - количество записанных амплитуд в i-м объекте. Само по себе множество объектов бесконечно, однако известны значения первых p элементов. Обозначим их через $\widehat{X} = \{x_0, ..., x_{p-1}\}$.

Пусть $Y = \{y_0, ..., y_{p-1}, ...\}$ - множество скалярных меток, а $\widehat{Y} = \{y_0, ..., y_{p-1}\}$ - множество известных скалярных меток для первых p объектов речи. Таким образом известно отображение $\widehat{Z} = \widehat{X} \to \widehat{Y}$, описываемое парами значений $\widehat{Z} = \{(x_0, y_0), ..., (x_{p-1}, y_{p-1})\}$.

Необходимо разработать алгоритм, который бы строил отображение $Z = X \to Y$.

Для того, чтобы решить поставленную задачу, необходимо разбить ее на следующие подзадачи и последовательно решить их:

- провести предобработку первоначальных данных, содержащих звуковой сигнал в виде наборов амплитуд;
- разработать алгоритм распознавания объектов речи;
- реализовать алгоритм распознавания объектов речи;
- провести вычислительные эксперименты и выяснить, какой метод решения задачи является наиболее эффективным.

В дальнейшем под объектом речи понимается отдельно взятое слово. В контексте этой работы словом является речевая команда, которая произносится на английском языке в микрофон. Базовый набор команд составляет необходимый минимум для управления программным интерфейсом медиаплеера. В рамках данной работы сам интерфейс не рассматривается.

Обзор литературы

Далее в хронологическом порядке описаны наиболее важные моменты и работы, связанные с поставленной задачей.

Первые шаги в распознавании речи были сделаны в 1952 году. Тогда трое исследователей Bell Labs - Стивен Балашек, Рулон Биддалф и Кей Дэвис - представили публике первый в истории аппарат Audrey, способный распознавать человеческую речь [1]. Это была система, позволявшая распознавать только цифры.

В 1957 году Фрэнк Розенблатт предложил математическую модель восприятия информации мозгом - персептрон. В своей статье [2], а позднее и в своей книге [3], он описал принципы работы модели. Этот момент можно считать появлением нейронных сетей как таковых. На сегодняшний день их развитие ушло вперёд, однако принципы, заложенные Розенблаттом, являются основополагающими в этой области.

В работе 1963 года [4], которую опубликовали Богерт, Хили и Тьюки, впервые описан кепстр для анализа геологических данных. Позднее этот подход будет использован в анализе речевого сигнала. В десятой главе книги Оппенгейма [8] описано его применение в распознавании речи.

В 1967 году впервые применяется спектр для анализа звуковых сигналов [5].

В 1971 году появилась одна из первых моделей распознавания большого словаря речевых команд. Она была представлена на конкурсе DARPA и носила название Harpy [7]. Методы, которые были использованы при её реализации актуальны и по сей день.

Для выделения речевых признаков впервые в 1980 году использован алгоритм мел-частотных кепстральных коэффициентов [9]. Он основан на кепстре, который Богерт, Хили и Тьюки использовали для геоанализа.

В 1995 году Ян ЛеКун предложил модель свёрточной нейронной сети

для распознавания изображений и речи [11]. Все современные архитектуры нейронных сетей в своём большинстве - модификации свёрточной сети.

Глава 1. Теоретические сведения

В этой главе рассмотрены теоретические сведения о человеческой речи, речевых признаках и общая схема алгоритма распознавания речи.

1.1 Речь как объект распознавания

Речь представляет собой акустическую волну, которая излучается системой органов: лёгкими, бронхами и трахеей, а затем преобразуется в голосовом тракте. Если предположить, что источники возбуждения и форма голосового тракта относительно независимы, то речевой аппарат человека можно представить в виде совокупности генератора тоновых сигналов и фильтров. На рисунке 1 изображена схема речеобразования.

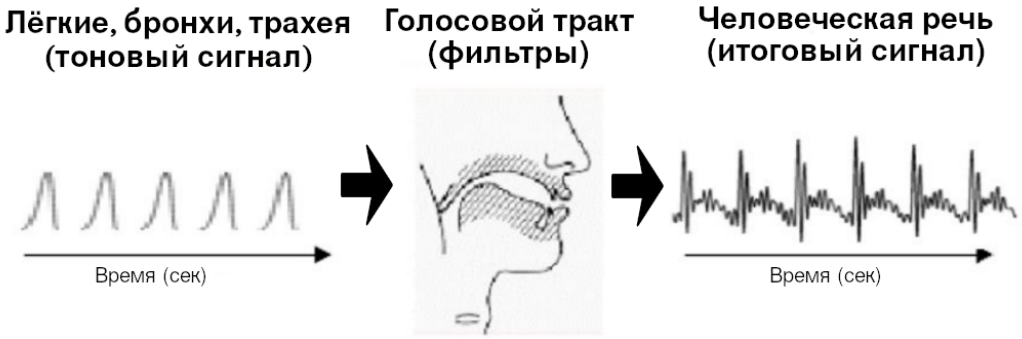

Рис. 1: Речеобразование

Распознавание речи происходит на основе анализа определённых признаков, которые причастны определенным звукам. За создание этих особенностей отвечает голосовой тракт человека. Поэтому одной из подзадач предобработки звука является выделение той части сигнала, которая была сформирована именно голосовым трактом.

Произнесённая речь поступает в приёмники звука. В случае человеческого уха происходит следующее. Звуковые волны проходят через наружное ухо в среднее и вызывают вибрацию барабанной перепонки. Колебания с

барабанной перепонки передаются на маленькие слуховые косточки в среднем ухе. А со слуховых косточек - во внутреннее ухо. Когда эти колебания достигают улитки, они воздействуют на специальные клетки - волосковые. Волосковые клетки преобразуют колебания в электрические нервные импульсы. Слуховой нерв соединяет улитку с центрами слуха в головном мозге. Когда электрические нервные импульсы достигают головного мозга, они воспринимаются как звук и обрабатываются.

В случае с компьютером и подключённым к нему микрофоном, принцип схожий. Звуковые волны преобразуются микрофоном в электрический сигнал, который далее обрабатывается АЦП в составе звуковой платы компьютера. На выходе - цифровые данные, подлежащие обработке в рамках поставленной задачи.

Речь можно представить в виде последовательности предложений, а их в свою очередь в виде последовательности слов. Слова состоят из фонем. В общем случае речь непрерывна, то есть слова не отделяются друг от друга паузами, за исключением требующих этого особенностей произносимой речи. В этой работе не рассматривается общий случай. Здесь рассмотрен частный случай, когда речь состоит из отдельных слов, отделяемых друг от друга тишиной. Под тишиной понимается не полное отсутствие звука, а временные промежутки, в течение которых говорящий молчит, делает паузы. Этот выбор был обусловлен командным типом системы распознавания, которая работает с отдельными словами.

1.2 Речь в компьютерном представлении

Каждая речевая команда с точки зрения звуковой записи в компьютере - это набор амплитудных значений, полученных с микрофона, преобразованных АЦП в числовые и записанных в звуковой файл формата wav.

1.3 Общая схема алгоритма распознавания

На рисунке 2 представлена общая схема работы алгоритма распознавания речевых команд. Алгоритм разделен на два основных блока, обозначенных на рисунке как Блок 1 и Блок 2. На вход алгоритму поступает набор амплитудных значений и соответствующая частота дискретизации в формате wav. На выходе алгоритма - текстовое представление команды.

- Блок 1 - блок, отвечающий за первоначальную обработку данных и приведение их к единому формату.
- Блок 2 - блок, отвечающий за классификацию унифицированных данных. Этот блок может работать в двух режимах: обучения и непосредственной работы. В режиме обучения происходит корректировка параметров алгоритма, которые влияют на конечный результат. В режиме непосредственной работы этого не происходит.

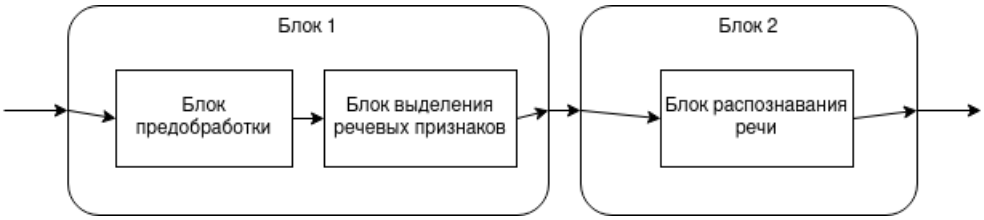

Рис. 2: Общая схема работы алгоритма распознавания речевых команд

Глава 2. Описание решения

В этой главе рассмотрен весь алгоритм распознавания речевых команд. Для удобства понимания названия параграфов расположены в том же порядке, что и этапы в самом алгоритме.

2.1 Предобработка

Каждая команда записана в звуковой wav файл. В каждом файле - набор амплитудных значений, которые были получены в результате записи команды дикторами.

2.1.1 Нормализация сигнала

Сначала проводится нормализация амплитуд. Каждое значение амплитуды приводится к такому значению, чтобы максимум среди всех амплитуд звуковой дорожки был равен единице по формуле:

$$\overline{x}_i = \frac{x_i}{\max_j |x_j|},\ i = \overline{0, p-1},\ j \in [0, p-1], \tag{1}$$

где x - значение амплитуды, $\overline{x}$ - новое значение амплитуды, p - количество амплитудных значений в звуковой дорожке.

Таким образом, все значения амплитуд принимают значения в диапазоне $[0, 1]$.

2.1.2 Удаление постоянной составляющей

Постоянная составляющая (DC-offset) - это смещение амплитуды сигнала на некоторую постоянную величину. Она возникает в цифровом сигнале, полученном с АЦП, из-за разницы напряжения между звуковой картой и устройством ввода. Данный эффект является помехой, от которой нужно избавиться. Для этого необходимо вычесть из каждого значения амплитуды

среднее арифметическое всех значений амплитуд по формуле:

$$\overline{x}_i = x_i - \sum_{j=0}^{p-1} x_j,\ i = \overline{0, p-1}, \tag{2}$$

где x - значение амплитуды полученное на этапе нормализации, $\overline{x}$ - новое значение амплитуды, p - количество амплитудных значений в звуковой дорожке.

Все $\overline{x}_i$ переобозначаются как x_i.

2.1.3 Выделение начальной и конечной точек слова

Каждая звуковая дорожка содержит в себе помимо фрагментов звукового сигнала (команды) ещё и фрагменты тишины. Очень важно отделить звуковой сигнал от фрагментов тишины, так как именно он несёт в себе всю информацию о команде.

Для того, чтобы выделить звуковой сигнал и «обрезать» тишину в начале и в конце записи, используется алгоритм, описанный в статье [6]. Каждая звуковая дорожка разбивается на фреймы - наборы амплитуд, каждый длительностью 20 мс. Начала фреймов расположены с периодичностью 10 мс. Таким образом, фреймы пересекаются между собой. Это обеспечивает целостность обработки звукового сигнала, позволяя обрабатывать важные фонемообразующие особенности.

Затем для каждого фрейма вычисляется мгновенная энергия:

$$E_k = \sum_{m=0}^{N-1} x_{k_m}^2,\ k = \overline{0, z-1}, \tag{3}$$

где z - количество фреймов для конкретной звуковой записи, N - длина одного фрейма (количество амплитуд в одном фрейме).

Вычисление функции мгновенной энергии имеет значительный недостаток. У неё велика чувствительность к относительно большим значениям амплитуды из-за возведения их во вторую степень. Это ведёт к искажению со-

отношений амплитудных значений звукового сигнала друг к другу. Поэтому функция мгновенной энергии переопределяется как:

$$E_k = \sum_{m=0}^{N-1} |x_{k_m}|,\ k = \overline{0, z-1}, \tag{4}$$

где z - количество фреймов для конкретной звуковой записи, N - длина одного фрейма.

После того, как посчитаны мгновенные энергии для каждого фрейма, вычисляются нижнее и верхнее пороговые значения:

$$\begin{aligned} I_1 &= 0.03 \cdot (MX - MN) + MN \\ I_2 &= 4 \cdot MN \\ ITL &= min(I_1,\ I_2) \\ ITU &= 10 \cdot ITL, \end{aligned} \tag{5}$$

где MN, MX - минимум и максимум мгновенной энергии среди всех фреймов соответственно, ITL, ITU - нижнее и верхнее пороговое значение.

Происходит поиск фрейма, с которого начинается слово, начиная с самого первого фрейма. Фрейм, в котором значение мгновенной энергии превышает ITL, предварительно помечается как начало слова. Затем, начиная с этого помеченного фрейма, происходит поиск фрейма, в котором значение мгновенной энергии превышает ITU. Если значение мгновенной энергии для какого-то фрейма во время последнего поиска меньше ITL, то этот фрейм становится предварительным началом слова.

Аналогично происходит поиск конца слова в звуковой дорожке, но поиск по фреймам происходит не с начала сигнала, а с конца.

После этого этапа имеются два предварительно помеченных фрейма m_{begin}, m_{end} - начало и конец слова в звуковом файле соответственно.

Функция мгновенной энергии, определённая формулой (4), хорошо

справляется с отделением звонких звуков от тишины. Но глухие она отделяет плохо. Поэтому используется вторая характеристика для доопределения начала и конца слова - число переходов через ноль. Это количество таких случаев, когда соседние значения амплитуд имеют противоположные знаки. Определяется формулой:

$$Z_k = \frac{1}{2}\sum_{m=1}^{N-1} |sgn(x_{k_{m-1}}) - sgn(x_{k_m})|,\ k = \overline{0, z-1}, \tag{6}$$

где z - количество фреймов для конкретной звуковой записи, N - длина одного фрейма.

Подразумевается, что первые 100 мс звуковой записи - это тишина, и речь начинается позднее.

Вычисляется среднее значение переходов через ноль в течение первых 100 мс (7) и среднее квадратическое отклонение количества переходов через ноль в течение первых 100 мс (8):

$$IZC = \frac{1}{z}\sum_{k=0}^{z-1} Z_k \tag{7}$$

$$\sigma_{IZC} = \sqrt{\frac{1}{z}\sum_{k=0}^{z-1}(Z_k - IZC)^2}, \tag{8}$$

где z - количество фреймов для конкретной звуковой записи. Затем вычисляется значение пороговой функции числа переходов через ноль по формуле:

$$IZCT = min(IF,\ IZC + 2\sigma_{IZC}), \tag{9}$$

где IF - фиксированное количество переходов через ноль. В данном случае оно составляет 25 пересечений за 10 мс, то есть $IF = 2.5$.

Далее происходит уточнение точек начала и конца слова в звуковой дорожке. Начиная от фрейма m_{begin} влево происходит поиск фреймов, у которых

число переходов через ноль выше порогового значения. Поиск происходит на расстоянии 25 фреймов, так как производится уточнение границ слова. Если пороговое значение было превышено 3 или более раз, то фрейм, где это произошло впервые, помечается как начало слова и обозначается как r_{begin}. Если пороговое значение было превышено менее 3-х раз, то метка m_{begin} переобозначается как r_{begin}.

Аналогично от фрейма m_{end} происходит поиск вправо для уточнения точки конца слова, которая обозначается как r_{end}.

В результате уточнения точек начала и конца слова имеются 2 помеченных фрейма - r_{begin}, r_{end}. Сигнал обрезается, и в нем остаётся только речевая команда в виде набора фреймов $[r_{begin}, ..., r_{end}]$. Количество получившихся фреймов обозначается как u, а сами фреймы перенумеруются следующим образом: $[r_0, ..., r_{u-1}]$.

2.2 Выделение речевых признаков

Для того, чтобы выделить речевые признаки, используется алгоритм мел-частотных кепстральных коэффициентов [9] (далее - MFCC). Он является одним из стандартных подходов к решению поставленной задачи. Состоит MFCC из нескольких шагов:

1. Для каждой звукового сигнала проделать шаги:
    a. Разбить сигнал на фреймы.
    b. Для каждого фрейма проделать шаги:
        I. Получить спектр сигнала.
        II. Составить набор мел-фильтров.
        III. Применить мел-фильтры к спектру сигнала.
        IV. Прологарифмировать результат, полученный на предыдущем шаге.

V. Применить дискретное косинусное преобразование к результату предыдущего шага.

c. Объединить MFCC векторы коэффициентов в матрицу.

2. Привести матрицы коэффициентов MFCC каждой звуковой дорожки к одной размерности. Для этого дополнить их нулями слева до необходимой длины. Унифицированная размерность матриц выбирается как максимальная среди всех.

Why do we do these things?

We will now go a little more slowly through the steps and explain why each of the steps is necessary.

An audio signal is constantly changing, so to simplify things we assume that on short time scales the audio signal doesn't change much (when we say it doesn't change, we mean statistically i.e. statistically stationary, obviously the samples are constantly changing on even short time scales). This is why we frame the signal into 20-40ms frames. If the frame is much shorter we don't have enough samples to get a reliable spectral estimate, if it is longer the signal changes too much throughout the frame.

The next step is to calculate the power spectrum of each frame. This is motivated by the human cochlea (an organ in the ear) which vibrates at different spots depending on the frequency of the incoming sounds. Depending on the location in the cochlea that vibrates (which wobbles small hairs), different nerves fire informing the brain that certain frequencies are present. Our periodogram estimate performs a similar job for us, identifying which frequencies are present in the frame.

The periodogram spectral estimate still contains a lot of information not required for Automatic Speech Recognition (ASR). In particular the cochlea can not discern the difference between two closely spaced frequencies. This effect becomes more pronounced as the frequencies increase. For this reason we take

clumps of periodogram bins and sum them up to get an idea of how much energy exists in various frequency regions. This is performed by our Mel filterbank: the first filter is very narrow and gives an indication of how much energy exists near 0 Hertz. As the frequencies get higher our filters get wider as we become less concerned about variations. We are only interested in roughly how much energy occurs at each spot. The Mel scale tells us exactly how to space our filterbanks and how wide to make them. See below for how to calculate the spacing.

Once we have the filterbank energies, we take the logarithm of them. This is also motivated by human hearing: we don't hear loudness on a linear scale. Generally to double the percieved volume of a sound we need to put 8 times as much energy into it. This means that large variations in energy may not sound all that different if the sound is loud to begin with. This compression operation makes our features match more closely what humans actually hear. Why the logarithm and not a cube root? The logarithm allows us to use cepstral mean subtraction, which is a channel normalisation technique.

The final step is to compute the DCT of the log filterbank energies. There are 2 main reasons this is performed. Because our filterbanks are all overlapping, the filterbank energies are quite correlated with each other. The DCT decorrelates the energies which means diagonal covariance matrices can be used to model the features in e.g. a HMM classifier. But notice that only 12 of the 26 DCT coefficients are kept. This is because the higher DCT coefficients represent fast changes in the filterbank energies and it turns out that these fast changes actually degrade ASR performance, so we get a small improvement by dropping them.

Так как на выходе алгоритма выделения начальной и конечной точек слова получается набор фреймов, то шаг разбиения сигнала на фреймы опускается.

2.2.1 Получение спектра сигнала

Спектр сигнала S_{k_m} в k-ом фрейме - результат дискретного преобразования Фурье:

$$S_{k_m} = \sum_{n=0}^{N-1} x_{k_n} \cdot e^{\overline{\frac{-2\pi i}{N}}^{mn}},\ m = \overline{0, N-1},\ k = \overline{0, u-1}, \tag{10}$$

где u - количество фреймов, получившееся после выделения начальной и конечной точек слова, N - длина одного фрейма, i - мнимая единица.

2.2.2 Мел-шкала и расчёт мел-фильтров

Мел - психофизическая единица высоты звука. Она описывает значимость конкретной частоты в человеческом восприятии. Популярные формулы для перевода Герц в мел и обратно описаны в книге [10] на стр. 150:

$$mel(hz) = 1127 \cdot ln(1 + \frac{hz}{700}) \tag{11}$$

$$hz(mel) = 700 \cdot (e^{\overline{\frac{mel}{1127}}} - 1) \tag{12}$$

На рисунке 3 изображено сравнение шкал мел и Гц.

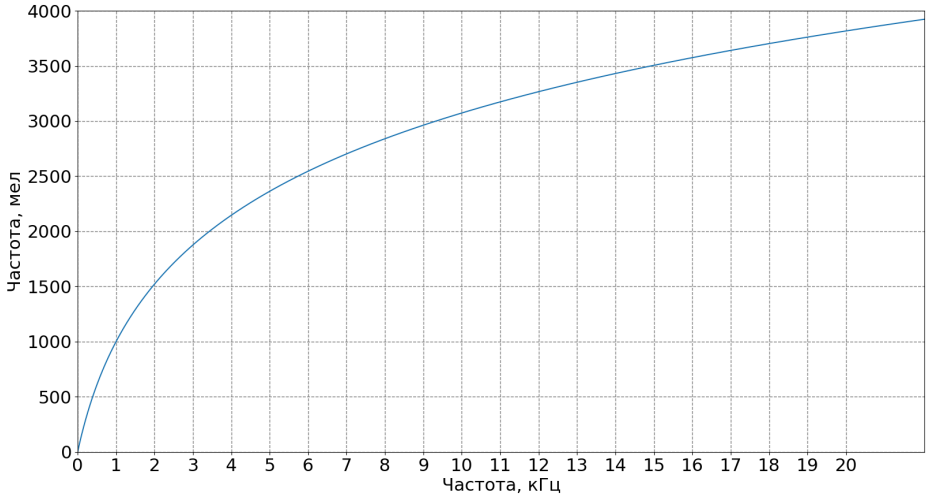

Рис. 3: Сравнение шкал мел и Гц

Составляются треугольные мел-фильтры в виде оконной функции:

$$H(v,\ b) = \begin{cases} 0, & b < f(v) \\ \dfrac{b - f(v)}{f(v+1) - f(v)}, & f(v) \le b < f(v+1) \\ \dfrac{f(v+2) - b}{f(v+2) - f(v+1)}, & f(v+1) \le b < f(v+2) \\ 0, & f(v+2) < b \end{cases}, \quad (13)$$

для которой f определяется как:

$$f(a) = \frac{N}{w} hz(mel(f_{min}) + a\frac{mel(f_{max}) - mel(f_{min})}{Q+1}), \quad (14)$$

где mel и hz - функции, определённые формулами (11) и (12) соответственно, w - частота дискретизации звуковой дорожки, $f_{min},\ f_{max}$ - нижний и верхний пороги частотного диапазона соответственно, N - длина одного фрейма, Q - количество мел-фильтров. Параметр Q обычно выбирается в диапазоне 20-40 (26 является стандартом).

На рисунке 4 в качестве примера изображена оконная функция для звуковой дорожки с параметрами $w = 44100$ Гц, $f_{min} = 0$ Гц, $f_{max} = 22050$

Гц, $N = 1024$, $Q = 13$.

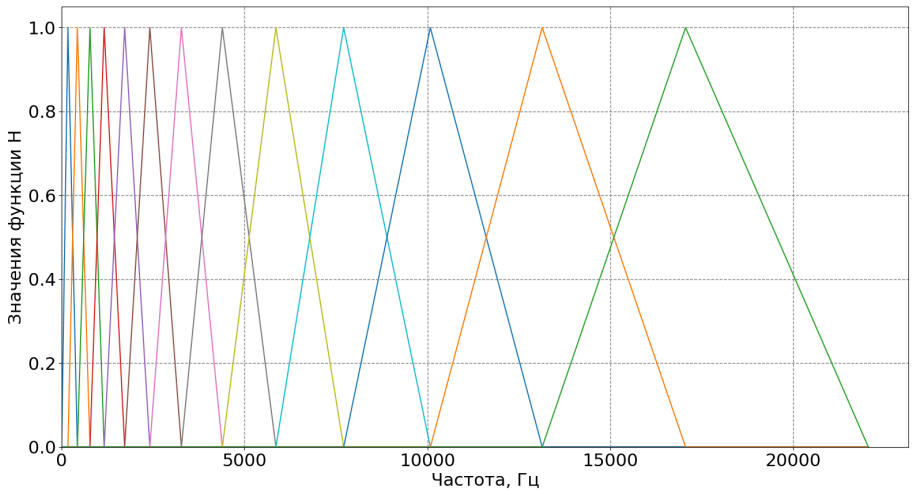

Рис. 4: Оконная функция

2.2.3 Получение кепстра сигнала

Кепстр - определяется в виде прямого преобразования Фурье от логарифма спектра мощности сигнала:

$$C(x(t)) = F(ln[F(x[t])]), \tag{15}$$

где $x(t)$ - входной сигнал, F - функция прямого преобразования Фурье.

Кепстр совместно с мел-фильтрами используется для выделения речевых признаков. Эта та часть звукового сигнала, которая была образована при помощи голосового тракта человека. В книге [8] можно найти описание того, как это происходит на стр. 367-380.

Выше описан кепстр для непрерывного сигнала. В случае с дискретным набором амплитуд в сигнале и учитывая мел-фильтры, $ln[F(x[t])$ записывается как:

$$W_{k_q} = ln(\sum_{m=0}^{N-1} |S_{k_m}|^2 \cdot H(q,m)),\ q = \overline{0, Q-1},\ k = \overline{0, u-1}, \tag{16}$$

где u - количество фреймов, получившееся после выделения начальной и конечной точек слова, Q - выбранное количество мел-фильтров.

В MFCC вместо прямого преобразования Фурье над логарифмом спектра мощности сигнала используется дискретное косинусное преобразование как его частный случай прямого преобразования Фурье. Получение итоговых коэффициентов MFCC происходит по формуле:

$$c_{k_n} = ln(\sum_{m=0}^{Q-1} W_{k_m} cos(\frac{\pi}{Q}(m+\frac{1}{2})n)),\ n = \overline{0, d-1}, d \leq Q,\ k = \overline{0, u-1}, \quad (17)$$

где u - количество фреймов, получившееся после выделения начальной и конечной точек слова, Q - выбранное количество мел-фильтров, d - выбранное количество коэффициентов MFCC (обычно $d = 12$), π - математическая константа.

Таким образом, строится матрица для каждой звуковой дорожки следующего вида:

$$C_{u \times d} = \begin{pmatrix} c_{00} & c_{01} & \cdots & c_{0(d-1)} \\ c_{10} & c_{11} & \cdots & c_{1(d-1)} \\ \vdots & \vdots & \ddots & \vdots \\ c_{(u-1)0} & c_{(u-1)1} & \cdots & c_{(u-1)(d-1)} \end{pmatrix}$$

где u - количество фреймов, получившееся после выделения начальной и конечной точек слова, d - выбранное количество коэффициентов MFCC.

2.2.4 Приведение данных к одной размерности

Среди всех значений u существует максимальное u_{max}. Для каждой матрицы, соответствующей определённой звуковой дорожке, проделаем следующее:

- если $u < u_{max}$, то соответствующая матрица C дополняется нулями

слева:

$$C_{u_{max} \times d} = \begin{pmatrix} 0 & \dots & 0 & c_{00} & c_{01} & \dots & c_{0(d-1)} \\ 0 & \dots & 0 & c_{10} & c_{11} & \dots & c_{1(d-1)} \\ \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \dots & 0 & c_{(u-1)0} & c_{(u-1)1} & \dots & c_{(u-1)(d-1)} \end{pmatrix}$$

- если $u = u_{max}$, то матрица C остаётся без изменений.

Таким образом, все матрицы приводятся к одной размерности $u_{max} \times d$.

2.3 Распознавание речевых команд

Распознавание речевых команд происходит при помощи применения технологии нейронных сетей. В данной работе рассматриваются два типа нейронных сетей: многослойный персептрон и свёрточная сеть. Производится сравнение производительности этих двух типов при разных параметрах обучения.

2.3.1 Многослойный персептрон

Этот вид нейронной сети описан в третьей части книги Фрэнка Розенблатта [3], который первый предложил модель персептрона. Она состоит их входного слоя, полносвязных слоев и выходного слоя.

2.3.2 Свёрточная нейронная сеть

Данный тип нейронной сети был предложен Яном ЛеКуном [11]. Это многослойная сеть, позволяющая обеспечить устойчивость распознавания к инвариантным изменениям данных за счёт общих весов и локального рецептивного поля.

Входной слой сети состоит из одной плоскости. Его размерность совпадает с размерностью входных данных.

Последующие слои - свёрточные. Каждый свёрточный слой состоит из нескольких плоскостей нейронов, которые известны как карты признаков. Каждый нейрон в свёрточном слое соединён с небольшой областью предыдущего слоя. В этом заключается принцип локального рецептивного поля.

После каждого свёрточного слоя, который получил локальные признаки, стоит пулинг слой. Его задача состоит в понижении размерности данных.

После всех свёрточных слоёв следует выпрямляющий слой, который преобразует данные к вектору.

Далее следуют полносвязные слои. Иногда добавляется дропаут слой, который отключает некоторые случайные нейроны в процессе обучения. Это позволяет бороться с переобучением сети.

Выходной слой имеет размерность требуемых выходных данных.

2.3.3 Входные и выходные данные модели

Входной тензор модели - матрица MFCC коэффициентов для соответствующей команды. Его размерность - $u_{max} \times d$. Для составленного датасета $u_max = 400, d = 13$.

Выходной тензор имеет размерность $g \times 1$, где g - количество возможных команд для распознавания. Для составленного датасета $g = 11$. Скалярная метка распознанной команды соответствует индексу максимального элемента выходного тензора. Индексация выходном тензоре начинается с 0.

2.3.4 Архитектура нейронной сети

Архитектура свёрточной нейронной сети приведена на рисунке 6. Как видно из рисунка, сеть включает в себя:

- Входной слой InputLayer (размерность входных данных - 400×13)
- Слой свёртки Conv2D (32 нейрона, размерность ядра сверки - 5×5, функция активации - ReLu)

- Слой пулинга AveragePooling2D (размерность пула - 2×2)
- Слой свёртки Conv2D (64 нейрона, размерность ядра сверки - 5×5, функция активации - ReLu)
- Слой пулинга AveragePooling2D (размерность пула - 2×2)
- Выпрямляющий слой Flatten
- Полносвязный слой Dense (128 нейронов, функция активации - ReLu)
- Слой дропаута Dropout (процент исключения случайных нейронов от общего числа в слое - 30%)
- Выходной полносвязный слой Dense (11 нейронов, функция активации - Softmax)

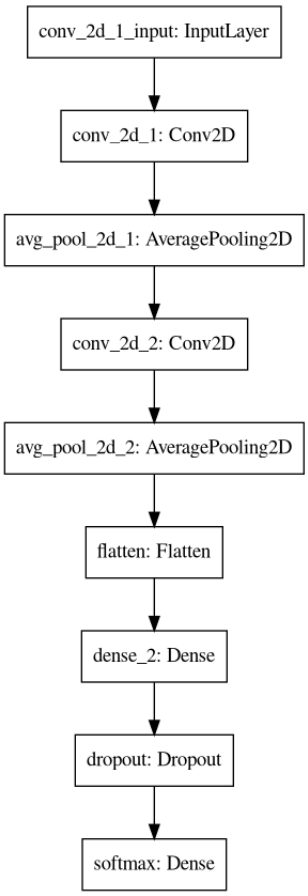

Рис. 5: Структура свёрточной нейронной сети

Архитектура свёрточной нейронной сети приведена на рисунке 6. Как видно из рисунка, сеть включает в себя:

- Входной слой InputLayer (размерность входных данных - 400×13)
- Выпрямляющий слой Flatten
- Полносвязный слой Dense (256 нейронов, функция активации - ReLu, регуляризатор ядра - L2 с параметром $\lambda = 0.00001$)
- Полносвязный слой Dense (128 нейронов, функция активации - ReLu, регуляризатор ядра - L2 с параметром $\lambda = 0.00001$)
- Полносвязный слой Dense (128 нейронов, функция активации - ReLu, регуляризатор ядра - L2 с параметром $\lambda = 0.00001$)
- Полносвязный слой Dense (64 нейрона, функция активации - ReLu, регуляризатор ядра - L2 с параметром $\lambda = 0.00001$)
- Выходной полносвязный слой Dense(11 нейронов, функция активации - Softmax)

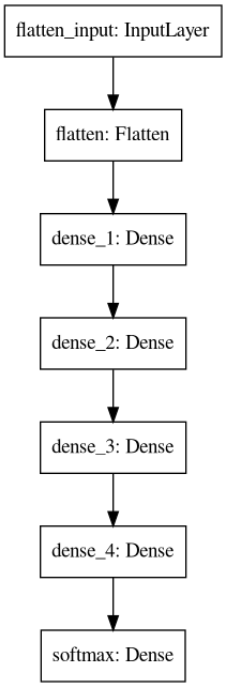

Рис. 6: Структура многослойного персептрона

2.4 Программная реализация

Построение программного интерфейса для блоков предобработки и распознавания команд было произведено при помощи языка программирования Python 3.8.

Блок предобработки был реализован при помощи библиотек numpy, scipy.

Блок распознавания был реализован при помощи библиотек jupyterlab, Keras, numpy. Для анализа результатов обучения нейронных сетей были использованы библиотеки pandas, matplotlib, seaborn.

Программный комплекс разделен на две основные части:

- Корневой файл «preprocessing.py», в котором реализована предобработка. Весь вспомогательный функционал вынесен в отдельный модуль «helpers».
- Корневой файл «boxy.ipynb», в котором реализовано обучение и тестирование нейронных сетей. Весь вспомогательный функционал также вынесен в отдельный модуль «helpers».

В корневой директории «recorded_audio» хранятся записанные шестью разными дикторами звуковые файлы, содержащие команды. Каждый файл содержит в своём названии индекс речевой команды, которая в нём записана.

В корневую директорию «data», в процессе предобработки файлов из директории «recorded_audio», сохраняются файлы:

- «speaker{i}_data.npy»[1]
- «speaker{i}_labels.npy»[1],

где $i = \overline{1,6}$ - номер диктора. Их описание приводится в разделе 2.5.

В корневой директории «logs» в процессе предобработки данных создаётся файл «log.log», в который записывается краткий отчёт об обработке каждой звукового файла.

В корневой директории «model_checkpoints» в процессе обучения нейронной сети создаются дочерние директории «experiment{i}»[1], $i = \overline{1,3}$ - номер эксперимента. В этой дочерней директории создаётся своя дочерняя директория «nn_type». Параметр nn_type принимает значения «cnn» при обучении свёрточной сети и «mlp» при обучении многослойного персептрона. В директорию «nn_type» сохраняются веса сети с наилучшей точностью распознавания на валидационных данных.

Для создания датасета было разработано веб-приложение при помощи языков Javascript, HTML, CSS и фреймворка NodeJS. Этот сервис позволяет записывать речевые команды и сохранять их в нужном для программы предобработки wav формате с правильным названием. Это позволило в короткие сроки записать необходимое количество дикторов и уменьшило скорость записи в несколько раз, так как в обычных программах записи звука очень много времени уходит на сохранение с правильными параметрами звуковой дорожки. Код веб-приложения доступен в приложении.

2.5 Описание датасета

Составленный датасет состоит из 11 команд, записанных шестью дикторами. Каждый диктор работал с командами : «back», «down», «menu», «off», «on», «open», «play», «power», «stop», «up», «volume». В таблице 1 указаны типы голосов дикторов и данные количестве записанных команд.

[1]Фигурные скобки не являются частью имени файла или директории.

Диктор	Тип голоса	Кол-во звук. дорожек на каждую команду	Сумм. кол-во звук. дорожек
speaker1	Мужской	50	550
speaker2	Мужской	40	440
speaker3	Мужской	40	440
speaker4	Мужской	40	440
speaker5	Мужской	50	550
speaker6	Женский	50	550

Таблица 1: Типы дикторов и данные количестве записанных команд

Датасет предварительно разделяется на тренировочную и тестовую части. На тренировочную часть отводится 70% данных каждого диктора, на тестовую часть - 30%.

Матрицы коэффициентов MFCC, полученных на этапе 2.2.3 объединяются в массив и записываются в файл «speaker{i}_data.npy»[1] в виде numpy массива. В файл «speaker{i}_labels.npy»[1] записываются скалярные метки команд в виде массива, в том же порядке, что и соответствующие им матрицы коэффициентов MFCC. Здесь $i = \overline{1,6}$ - номер диктора. Соответствие произнесённых диктором команд и их скалярных меток представлено в таблице 2.

Команда	back	down	menu	off	on	open	play	power	stop	up	volume
Скалярная метка	0	1	2	3	4	5	6	7	8	9	10

Таблица 2: Типы дикторов и данные количестве записанных команд

[1] Фигурные скобки не являются частью имени файла или директории.

Глава 3. Вычислительные эксперименты

Обозначения, которые используются далее:
all_speakers = [speaker1, speaker2, speaker3, speaker4, speaker5, speaker6]
all_male_speakers = [speaker1, speaker2, speaker3, speaker4, speaker5]

3.1 Описание экспериментов

Для каждого из двух типов нейронных сетей, многослойного персептрона и свёрточной сети, проводится три эксперимента:

1. нейронная сеть обучается на первом дикторе с мужским голосом, тестирование производится сначала на каждом дикторе, а потом на всех вместе

2. нейронная сеть обучается на всех дикторах с мужским голосом, тестирование производится сначала на каждом дикторе, а потом на всех вместе

3. нейронная сеть обучается на всех дикторах, тестирование производится сначала на каждом дикторе, а потом на всех вместе.

Обучение производится со следующими параметрами:

- после каждой эпохи тренировочные данные перемешиваются
- 15% тренировочных данных в каждой эпохе - валидационные
- метрика для оценки эффективности - точность (в библиотеке Keras называется accuracy)
- функция потерь - категориальная кросс-энтропия (categorical_crossentropy)
- алгоритм оптимизации - Adam
- количество эпох обучения сети - 50

- ранняя остановка обучения сети, если в течение 20 эпох значение функции потерь на валидационных данных не улучшается.

В случае обучения на all_speakers, помимо тестирования строится матрица путаницы (confusion matrix) для каждого из четырёх пороговых значений: 0.5, 0.6, 0.7, 0.8. Анализируя её, можно подобрать оптимальное пороговое значение для более удобного использования интерфейса распознавания речевых команд уже при его внедрении в программное обеспечение медиаплеера.

Смысл порогового значения следующий.

- Если максимальный элемент в выходном тензоре нейронной сети ниже порогового значения, то команда распознаётся как тишина. В этом случае интерфейс медиаплеера просит повторить команду ещё раз.

- Если максимальный элемент в выходном тензоре нейронной сети выше или равен пороговому значению, то команда распознаётся как соответствующая индексу этого максимального элемента.

3.2 Результаты экспериментов

Графики обучения многослойного персептрона приведены на рисунках 7, 8, 9.

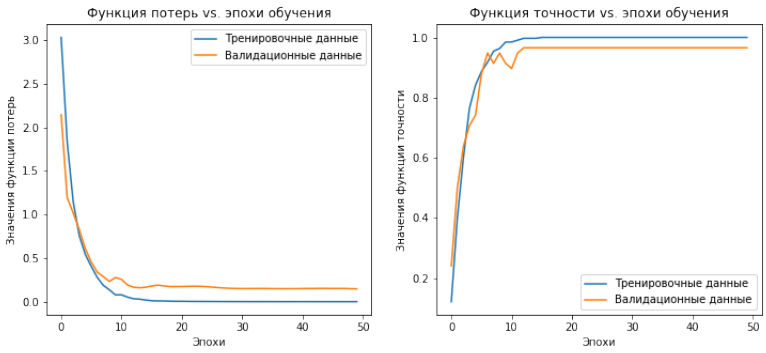

Рис. 7: Графики функций потерь и точности многослойного персептрона в течение обучения на speaker1

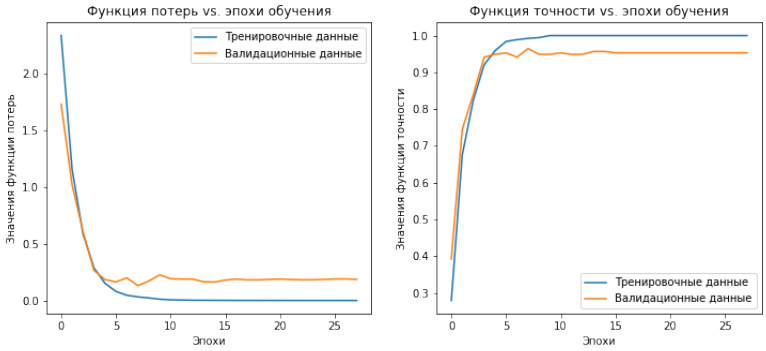

Рис. 8: Графики функций потерь и точности многослойного персептрона в течение обучения на all_male_speakers

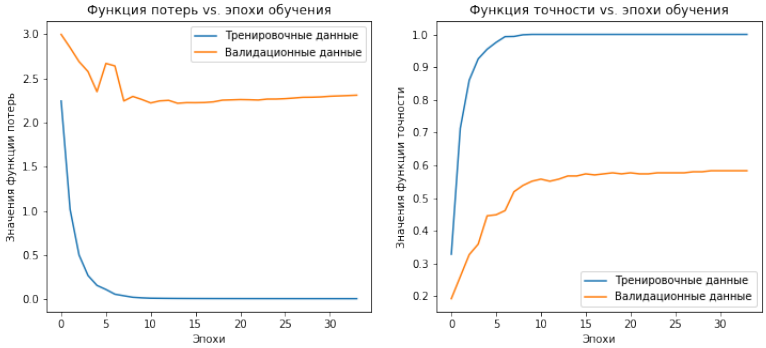

Рис. 9: Графики функций потерь и точности многослойного персептрона в течение обучения на all_speakers

Графики обучения свёрточной сети приведены на рисунках 10, 11, 12.

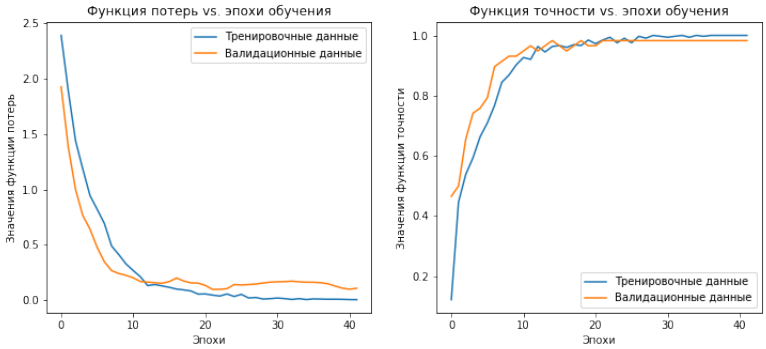

Рис. 10: Графики функций потерь и точности свёрточной сети в течение обучения на speaker1

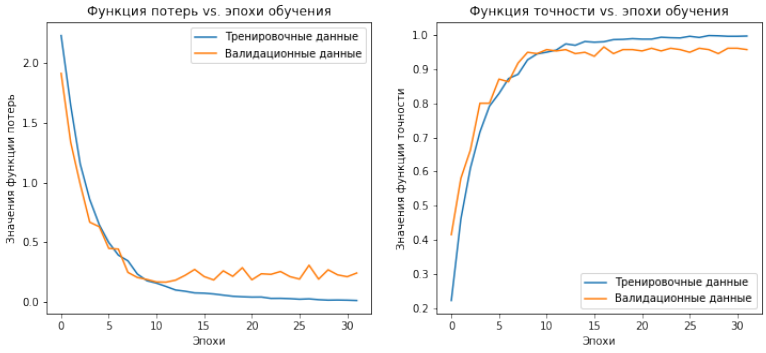

Рис. 11: Графики функций потерь и точности свёрточной сети в течение обучения на all_male_speakers

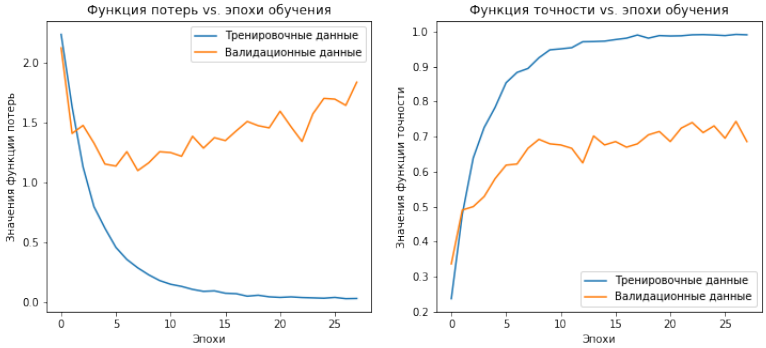

Рис. 12: Графики функций потерь и точности свёрточной сети в течение обучения на all_speakers

Матрицы путаницы для многослойного персептрона при обучении на all_speakers представлены на рисунке 13.

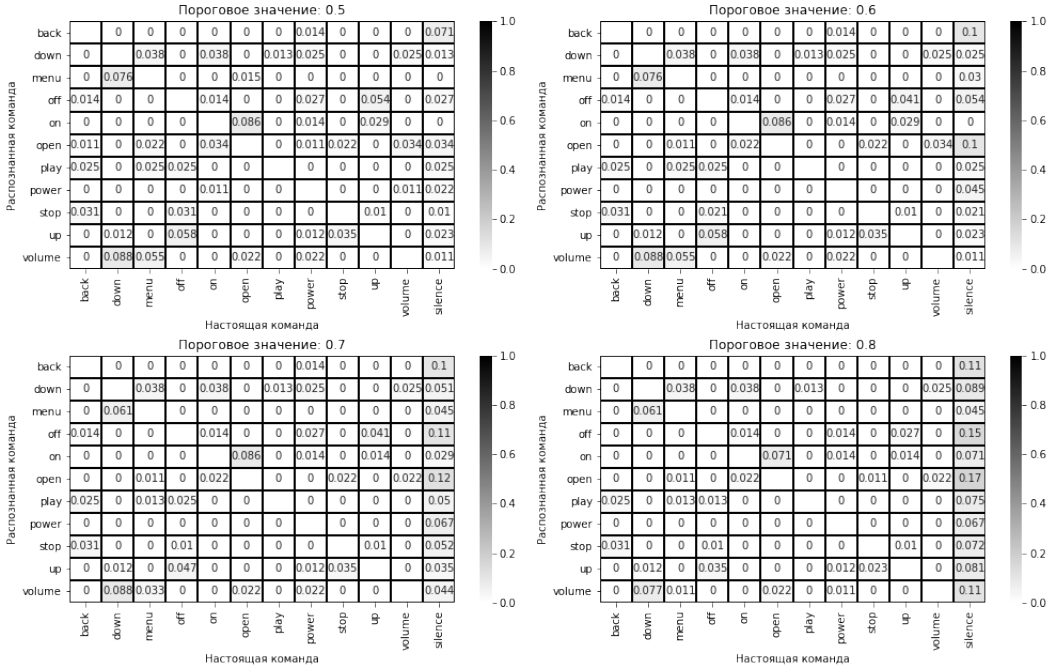

Рис. 13: Матрицы путаниц для многослойного персептрона при обучении на all_speakers

Матрицы путаницы для свёрточной сети при обучении на all_speakers представлены на рисунке 14.

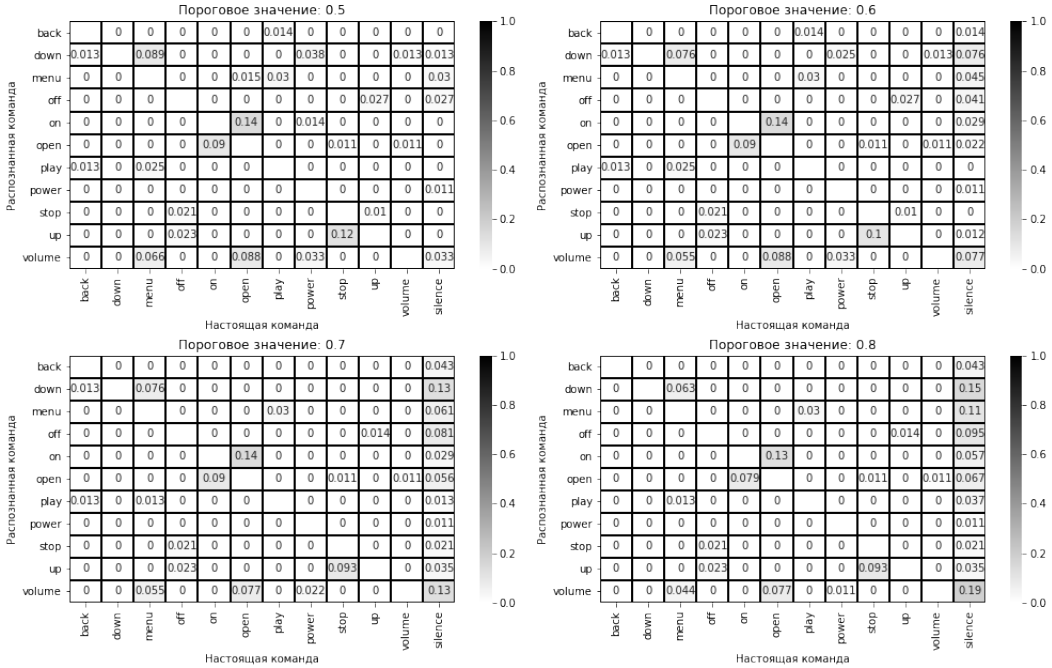

Рис. 14: Матрицы путаницы для свёрточной сети при обучении на all_speakers

Результаты тестирования представлены в таблице 3.

train_data	test_speaker	cnn_loss	mlp_loss	cnn_accuracy	mlp_accuracy
speaker1	speaker1	0.149	0.129	0.976	0.97
speaker1	speaker2	15.229	7.17	0.136	0.136
speaker1	speaker3	4.424	2.578	0.439	0.394
speaker1	speaker4	12.056	6.421	0.091	0.25
speaker1	speaker5	2.976	3.694	0.6	0.612
speaker1	speaker6	16.2	9.643	0.182	0.145
speaker1	all_speakers	8.276	4.889	0.424	0.435
all_male_speakers	speaker1	0.054	0.08	0.994	0.976
all_male_speakers	speaker2	0.103	0.247	0.97	0.947
all_male_speakers	speaker3	0.382	0.278	0.947	0.97
all_male_speakers	speaker4	0.487	0.527	0.902	0.909
all_male_speakers	speaker5	0.273	0.242	0.958	0.952
all_male_speakers	speaker6	5.176	8.917	0.424	0.224
all_male_speakers	all_speakers	1.163	1.867	0.857	0.817
all_speakers	speaker1	0.081	0.145	0.976	0.97
all_speakers	speaker2	0.116	0.26	0.962	0.932
all_speakers	speaker3	0.273	0.216	0.955	0.962
all_speakers	speaker4	0.764	0.543	0.879	0.902
all_speakers	speaker5	0.006	0.011	1.0	1.0
all_speakers	speaker6	2.118	2.301	0.691	0.564
all_speakers	all_speakers	0.579	0.606	0.908	0.883

Таблица 3: Результаты тестирования

Выводы

Видно, что при обучении только на одном дикторе с мужским голосом, точность распознавания на всех остальных дикторах по отдельности низкая.

При обучении на всех дикторах с мужским голосом, распознавание на дикторе с женским голосом работает лучше, чем при обучении на одном дикторе с мужским голосом, но точность все-равно низкая.

При обучении на всех дикторах, распознавание на каждом даёт приемлемую точность. Однако стоит отметить, что если большая часть дикторов в тренировочной части имеет мужские голоса, то на дикторе с женским голосом распознавание работает хуже, чем на дикторах с мужским голосом.

Можно сделать вывод о том, что свёрточная нейронная сеть немного лучше справляется с поставленной задачей, чем многослойный персептрон. При этом количество весов, влияющее на время распознавания и на объем занимаемой памяти для их хранения, у свёрточной нейронной сети намного меньше из-за особенности её архитектуры.

Предложения по улучшению предложенной модели распознавания речи:

- увеличить размер датасета для обучения
- повысить глубину предложенных архитектур нейронных сетей
- увеличить количество коэффициентов MFCC в алгоритме выделения речевых признаков
- изменить количество мел-фильтров в алгоритме выделения речевых признаков.

Заключение

В данной работе:

- Проведена предобработка звуковых дорожек, содержащих команды в wav файлах
- Разработан алгоритм распознавания речевых команд
- Реализован алгоритм распознавания речевых команд
- Проведены вычислительные эксперименты, в результате которых показана работоспособность и эффективность работы алгоритма распознавания речевых команд.

Список литературы

[1] Davis K. N., Biddulph R., Balashek S. Automatic recognition of spoken digits // The Journal of the Acoustical Society of America, 1952. Vol. 24, No 6. P. 637-642

[2] Rosenblatt, F. The perceptron: A probabilistic model for information storage and organization in the brain // Psychological Review, 1958, Vol. 65(6), P. 386–408

[3] Rosenblatt, F. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan Books, Washington DC, 1961

[4] Bogert B. P., Healy M. J. R., Tukey J. W. The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking // Proceedings of the Symposium on Time Series Analysis, 1963, Ch. 15, P. 209-243

[5] Plomp R., Pols L. C. W., van der Geer J.P. Dimensional analysis of vowel spectra // The Journal of the Acoustical Society of America, 1967, Vol. 41, P. 707-712

[6] Rabiner L.R., Sambur, M.R. An Algorithm for Determining the Endpoints of Isolated Utterances // The Bell System Technical Journal, 1975, Vol. 54, No. 2, P. 297-315

[7] Newell A. Harpy, production systems and human cognition // Research Showcase @ Carnegie Mellon University, 1978

[8] Оппенгейм А. В., Шафер Р. В. Цифровая обработка сигналов / Под ред. С. Я. Шаца, М.: Связь, 1979. С. 416

[9] Davis S., Mermelstein P. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences // IEEE

Transactions on Acoustics, Speech, and Signal Processing, 1980, Vol. ASSP-28, No. 4, P. 357-366

[10] O'Shaughnessy D. Speech communication: human and machine, Addison-Wesley, 1987, P. 568

[11] LeCun Y., Bengio Y.. Convolutional networks for images, speech, and time-series // The Handbook of Brain Theory and Neural Networks, 1995, MIT

Приложение

Ссылка на репозиторий с программой веб-сервисом для записи датасета, состоящего из звуковых файлов: `https://gitlab.com/polotent/commandrecorder`

Ссылка на репозиторий с программой предобработки данных, обучением и тестированием нейронной сети: `https://gitlab.com/polotent/boxy`