Санкт–Петербургский государственный университет

Направление: 01.03.02 «Прикладная математика и информатика»

ООП: Прикладная математика, фундаментальная информатика и программирование

ОТЧЕТ О НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ ПРАКТИКЕ

Тема задания : Разработка системы распознавания речевых команд при помощи методов машинного обучения

Выполнил : Мирошниченко Александр Сергеевич группа 17.Б05

Руководитель практики от СПбГУ : Козынченко В. А., кандидат физ.-мат. наук, доцент

Санкт-Петербург

2021 г.

Содержание

Введение

Распознавание речи является одной из важнейших задач взаимодействия человека и компьютера на сегодняшний день. Одним из способов решения задачи распознавания речи является ввод информации в вычислительное устройство при помощи микрофона; далее происходит преобразование звуковых колебаний в электрический сигнал, оцифровываемый компьютером и записываемый в определенный звуковой формат в виде набора числовых данных. Полученное числовое представление преобразованного звукового сигнала можно обрабатывать при помощи алгоритма для достижения поставленных целей. Одной из таких целей является преобразование в текст - набор слов, произнесенных человеком при записи звука в микрофон и последующее выполнение компьютером или исполнительными устройствами действий, согласно заданным условиям соответствия действия и полученного слова-команды.

Данная проблема была актуальна с времен появления компьютеров и остается таковой и по сей день. Изначально для решения данной задачи применялись такие алгоритмы, как скрытые Марковские модели, методы динамического программирования, методы дискриминантного анализа, основанные на Байесовской дискриминации и другие. В последнее время стали актуальны методы, основанные на нейронных сетях. С ростом вычислительной мощности компьютеров стало возможным выполнение большего количества математических операций и популярность нейронных сетей, требующих такой мощности, растет.

Постановка задачи

Была поставлена задача создания алгоритма распознавания звуковых команд в виде отдельных слов при помощи нейронной сети.

Выбор инструментов для решения задачи

Для решения задачи был выбран язык программирования Python 3.8. Предобработку данных было решено реализовывать при помощи Python 3.8. В качестве библиотеки для реализации нейронной сети была выбрана библиотека Keras, включенная в библиотеку Tensorflow 2.4.1.

Для создания датасета был разработан веб-сервис на NodeJS, Javascript, HTML, CSS, который позволяет записывать команды и сохранять их в нужном для программы предобработки формате. Также это позволило записать необходимое количество дикторов, которые смогли довольно быстро наговорить команды.

Результаты вычислений

Было проведено 3 вычислительных эксперемента для нейронной сети типа CNN. Структура сети приведена на рисунке 1.

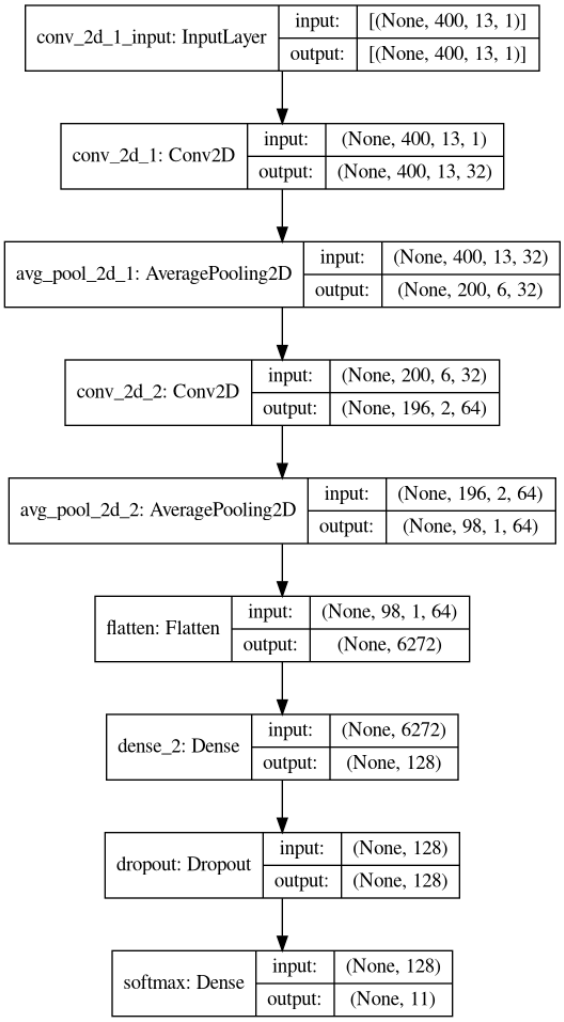

Рис. 1: Структура модели нейронной сети типа CNN

Все звуковые файлы были предобработаны при помощи алгоритма MFCC. Звуковая дорожка делится на фреймы. Каждый фрейм - отрезок звуковой дорожки длительностью 20 мс. Каждый фрейм начинается с момента (10 мс. $\times$ номер_фрейма), нумерация начинается с 0. Для каждой звуковой дорожки количество фреймов - 400. Если количество фреймов у дорожки меньше 400, то слева и справа добавляются нули. Это число было выбрано как максимально возможное количество фреймов для всех дорожек. Количество коэффициентов в алгоритме MFCC - 13, количество фильтров - 26. В итоге размерность данных, поступающих на вход нейронной сети - 400×13.

Датасет состоит из 6 дикторов. Каждый диктор записал 11 команд : 'back', 'down', 'menu', 'off',

'on', 'open', 'play', 'power', 'stop', 'up', 'volume'.

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

Первый эксперимент: нейронная сеть обучается на первом дикторе с мужским голосом, тестирование производится на каждом дикторе.

Второй эксперимент: нейронная сеть обучается на всех дикторах с мужским голосом, тестирование производится на каждом дикторе.

Третий эксперимент: нейронная сеть обучается на всех дикторах, тестирование производится на каждом дикторе.

Датасет предварительно разделяется на тренировочную и тестовую части. На тренировочную часть отводится 70% данных диктора, на тестовую часть - 30%. В процессе тренировки после каждой эпохи тренировочные данные перемешиваются. 15% тренировочных данных в каждой эпохе - валидационные. В качестве метрики для оценки эффективности была выбрана метрика точности (accuracy), а для валидации - функция потерь категориальной кросс-энтропии (val_loss). Алгоритм оптимизации - Adam. Максимальное количество эпох - 50. Если значение метрики val_loss не уменьшается в течение 20 эпох, то обучение останавливается.

Графики обучения для каждого из экспериментов приведены на рисунках 2, 3, 4.

all_speakers = [speaker1, speaker2, speaker3, speaker4, speaker5, speaker6]

all_male_speakers = [speaker1, speaker2, speaker3, speaker4, speaker5]

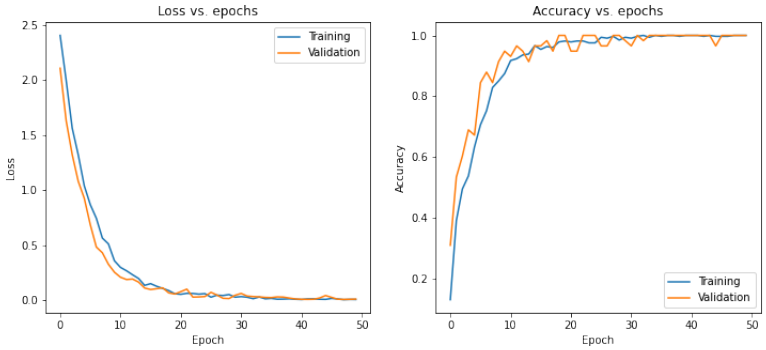

Рис. 2: Графики функции потерь и точности в течение обучения на speaker1

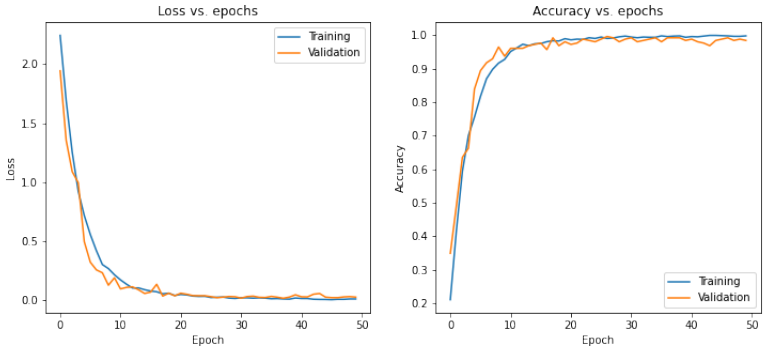

Рис. 3: Графики функции потерь и точности в течение обучения на all_male_speakers

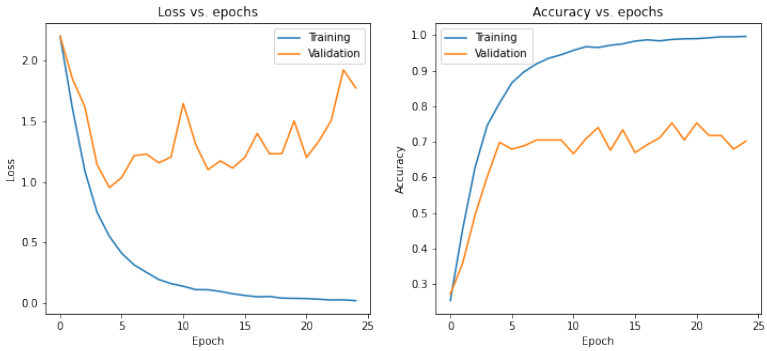

Рис. 4: Графики функции потерь и точности в течение обучения на all_speakers

train_data	test_speaker	cnn_loss	mlp_loss	cnn_accuracy	mlp_accuracy
speaker1	speaker1	0.056	0.11	0.994	0.97
speaker1	speaker2	10.015	7.371	0.167	0.136
speaker1	speaker3	4.832	2.438	0.341	0.402
speaker1	speaker4	8.05	6.128	0.197	0.265
speaker1	speaker5	2.526	3.015	0.618	0.606
speaker1	speaker6	23.892	7.414	0.145	0.2
all_male_speakers	speaker1	0.062	0.05	0.994	0.982
all_male_speakers	speaker2	0.252	0.462	0.962	0.864
all_male_speakers	speaker3	0.092	0.113	0.977	0.97
all_male_speakers	speaker4	0.846	0.525	0.909	0.909
all_male_speakers	speaker5	0.028	0.041	0.982	0.988
all_male_speakers	speaker6	10.637	9.418	0.333	0.261
all_speakers	speaker1	0.058	0.112	0.988	0.982
all_speakers	speaker2	0.153	0.539	0.947	0.833
all_speakers	speaker3	0.085	0.089	0.985	0.977
all_speakers	speaker4	0.561	0.938	0.909	0.902
all_speakers	speaker5	0.015	0.037	0.988	0.988
all_speakers	speaker6	2.144	3.601	0.679	0.588

Таблица 1: Результаты вычислений

В конце эксперимента помимо тестирования производится построение матрицы ошибок (confusion matrix) для каждого диктора и для каждого из четырех пороговых значений: 0.5, 0.6, 0.7, 0.8.

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] Aurélien G. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems / Aurélien G. — 2nd Edition — O'Reilly Media, 2019.

[2] Hendrik Purwins, Bo Li, Tuomas Virtanen, Jan Schlüter, Shuo-yiin Chang, Tara Sainath Deep Learning for Audio Signal Processing // Journal of Selected Topics of Signal Processing, Vol. 13, No. 2, May 2019,pages 206–219.

[3] Документация TensorFlow [Электронный ресурс]. — Режим доступа: https://www.tensorflow.org/api_docs/python/tf

[4] Портал ML Glossary [Электронный ресурс]. — Режим доступа: https://ml-cheatsheet.readthedocs.io

[5] Курс на платформе Coursera [Электронный ресурс]. — Режим доступа: https://www.coursera.org/learn/getting-started-with-tensor-flow2

[6] Страница на Википедии [Электронный ресурс]. — Режим доступа: https://en.wikipedia.org/wiki/Fourier_transform

[7] Страница на Википедии [Электронный ресурс]. — Режим доступа: https://en.wikipedia.org/wiki/Mel_scale

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`