МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

Кафедра математического обеспечения и применения ЭВМ

ОТЧЕТ

по лабораторной работе №3

по дисциплине «Анализ звука и голоса»

Тема: Классификация произвольных звуков

Студентка гр. 6304 _____ Вероха В.Н.

Преподаватель _____ Рыбин С.В.

Санкт-Петербург

2021

Цель работы.

Классификация произвольных аудиофайлов (набор данных “Freesound General-Purpose Audio”).

Описание данных на платформе “Kaggle”.

Некоторые звуки отчетливы и мгновенно узнаваемы, например: детский смех, бренчание гитары. Другие звуки нечеткие, и их сложно определить.

Кроме того, следует отметить, что чаще воспринимается смесь звуков, которые создают атмосферу. Например: шум строителей; шум транспорта за дверью, смешанный с громким смехом из комнаты; тиканье часов на стене. Отчасти, из-за огромного количества звуков не существует надежных автоматических универсальных систем тегов аудио. В настоящее время требуется много ручных усилий для таких задач, как аннотирование коллекций звуков и предоставление титров для неречевых событий в аудиовизуальном контенте. Чтобы решить эту проблему, Freesound (инициатива MTG-UPF, которая поддерживает совместную базу данных с более чем 370000 звуков, лицензированных Creative Commons) и команда Google Research Machine Perception (создатели AudioSet, крупномасштабного набора данных, вручную аннотированных звуковых событий с более чем 500 классов) объединились для разработки набора данных “Freesound General-Purpose Audio”.

Выполнение работы.

1. Подключены необходимы библиотеки. Результаты на рис. 1.

```
# Подключение библиотек

from sklearn.model_selection import train_test_split
from keras.models import Model
from keras.layers import Dense, Conv2D, BatchNormalization, Dropout, Input, GlobalAvgPool2D, GlobalMaxPool2D, concatenate
from tensorflow.keras.optimizers import Adam
from IPython.display import Audio
import wave
from joblib import Parallel, delayed
import librosa
from functools import partial
import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
```

Рисунок 1 — Подключение библиотек

2. Загружены данные “train.csv”. Содержимое данных представлено на рис. 2.

	fname	label	manually_verified
0	00044347.wav	Hi-hat	0
1	001ca53d.wav	Saxophone	1
2	002d256b.wav	Trumpet	0
3	0033e230.wav	Glockenspiel	1
4	00353774.wav	Cello	1

Рисунок 2 — Содержимое файла “train.csv”

3. Найдена длительность аудиофайлов и добавлена в датафрейм столбцом “duration”. Результаты представлены на рис. 3.

	fname	label	manually_verified	duration
0	00044347.wav	Hi-hat	0	14.00
1	001ca53d.wav	Saxophone	1	10.32
2	002d256b.wav	Trumpet	0	0.44
3	0033e230.wav	Glockenspiel	1	8.00
4	00353774.wav	Cello	1	4.52

Рисунок 3 — Расчет продолжительности аудиофайлов

4. Извлечено MFCC из аудио с использованием Librosa. В отчете для примера приведено извлечения из файла “0b82b3a5.wav”. Результаты представлены на рис. 4.

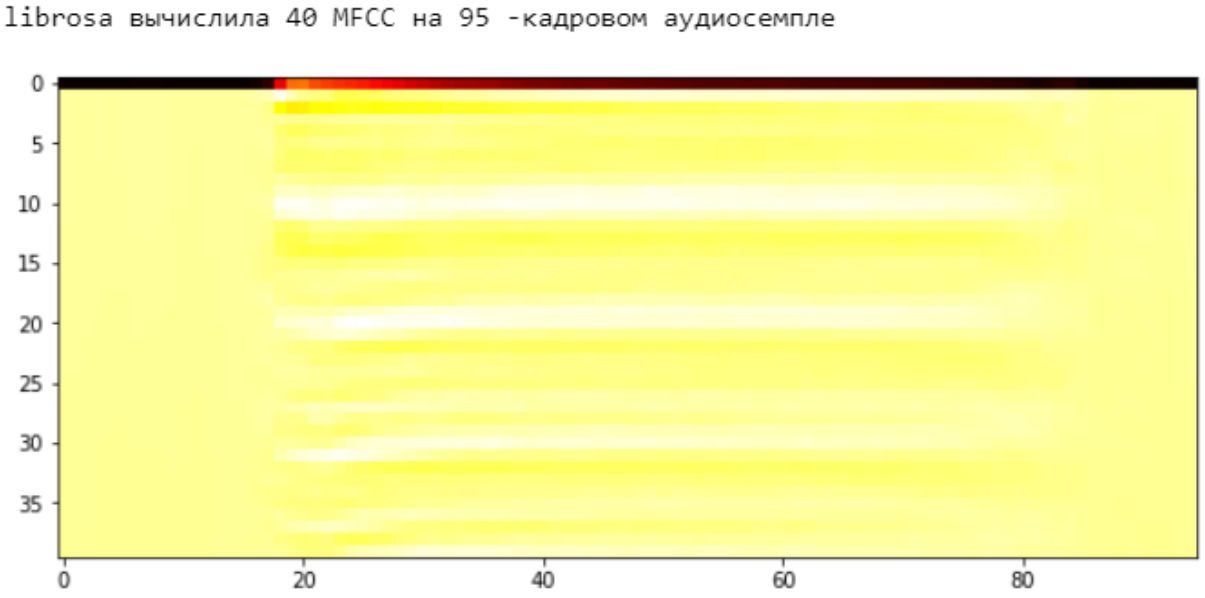

Рисунок 4 — Извлечение MFCC из аудио с использованием Librosa

5. Полученные данные из п.4 преобразованы для обучения. Результаты представлены на рис. 5.

```
Out[13]: array([[[-4.82412628e+02],
                 [-4.07738037e+02],
                 [-3.85441376e+02],
                 ...,
                 [ 0.00000000e+00],
                 [ 0.00000000e+00],
                 [ 0.00000000e+00]],

                [[ 9.47219391e+01],
                 [ 9.75328064e+01],
                 [ 1.02200790e+02],
```

Рисунок 5 — Вектор x_train для обучения

6. Составлен вектор данных с ярлыками возможных категорий данных. Результаты представлены на рис. 6.

Out[15]:

	fname	label	manually_verified	duration	label_number
2	002d256b.wav	Trumpet	0	0.44	0
6	003da8e5.wav	Knock	1	1.36	1
7	0048fd00.wav	Gunshot_or_gunfire	1	1.04	2
10	006f2f32.wav	Hi-hat	1	1.68	3
12	00780200.wav	Snare_drum	0	1.12	4

Рисунок 6 — Вектор y_train для обучения

7. Данные для моделирования разделены на выборки для обучения и тестов в соотношении 80/20.

8. Создана модель, архитектура которой представлена на рис. 7.

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 40, 130, 1)]	0	[]
batch_normalization (BatchNormalization)	(None, 40, 130, 1)	4	['input_1[0][0]']
conv2d (Conv2D)	(None, 40, 130, 10)	90	['batch_normalization[0][0]']
batch_normalization_1 (BatchNormalization)	(None, 40, 130, 10)	40	['conv2d[0][0]']
dropout (Dropout)	(None, 40, 130, 10)	0	['batch_normalization_1[0][0]']
conv2d_1 (Conv2D)	(None, 20, 65, 20)	1800	['dropout[0][0]']
batch_normalization_2 (BatchNormalization)	(None, 20, 65, 20)	80	['conv2d_1[0][0]']
dropout_1 (Dropout)	(None, 20, 65, 20)	0	['batch_normalization_2[0][0]']
conv2d_2 (Conv2D)	(None, 10, 33, 50)	9000	['dropout_1[0][0]']
batch_normalization_3 (BatchNormalization)	(None, 10, 33, 50)	200	['conv2d_2[0][0]']
dropout_2 (Dropout)	(None, 10, 33, 50)	0	['batch_normalization_3[0][0]']
conv2d_3 (Conv2D)	(None, 5, 17, 100)	45000	['dropout_2[0][0]']
batch_normalization_4 (BatchNormalization)	(None, 5, 17, 100)	400	['conv2d_3[0][0]']
dropout_3 (Dropout)	(None, 5, 17, 100)	0	['batch_normalization_4[0][0]']
global_average_pooling2d (GlobalAveragePooling2D)	(None, 100)	0	['dropout_3[0][0]']
global_max_pooling2d (GlobalMaxPooling2D)	(None, 100)	0	['dropout_3[0][0]']
concatenate (Concatenate)	(None, 200)	0	['global_average_pooling2d[0][0]', 'global_max_pooling2d[0][0]']
dense (Dense)	(None, 1000)	200000	['concatenate[0][0]']
dropout_4 (Dropout)	(None, 1000)	0	['dense[0][0]']
dense_1 (Dense)	(None, 41)	41041	['dropout_4[0][0]']

Total params: 297,655
Trainable params: 297,293
Non-trainable params: 362

Рисунок 7 — Архитектура модели

9. Точность построенной модели представлена на рис. 8.

```
Точность обучения: 98.86%
Точность тестирования: 71.95%
```

Рисунок 8 — Точность построенной модели

Выводы.

В результате проделанной лабораторной работы были получены навыки программирования на языке Python. Изучена задача классификации произвольных звуков на наборе данных “Freesound General-Purpose Audio”.

Создана модель с точностью обучения — 98.86%, тестирования — 71.95%.