|  |  |
| --- | --- |
| **СОГЛАСОВАНО**  Доцент кафедры  ИАНИ ННГУ, к.ф.-м.н.  \_\_\_\_\_\_\_\_\_\_\_\_\_ С.М. Липкин  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. | **УТВЕРЖДАЮ**  Профессор кафедры  ИАНИ ННГУ, д.т.н.  \_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Старостин  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г. |

**Пояснительная записка**

**«Разработка модели нейронной сети для определения присутствия дронов на звуковой дорожке»**

**НИР «Drone sound detection»**

**(Шифр ПО «AFR»)**

Ответственный исполнитель

\_\_\_\_\_\_\_\_\_\_\_\_ Хромов Н.А.

«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2020 г.

**Н. Новгород 2019**

Содержание

1. Общее описание алгоритма3

1.1 Назначение алгоритма3

1.2 Теоретическое описание алгоритма3

2. Реализация алгоритма3

2.1 Описание компонентов алгоритма3

2.2 Детали реализации алгоритма4

**3. Тестирование5**

3.1 Методы проведения тестирования5

3.1 Результаты тестирования5

**4. Интеграция с другими компонентами системы5**

5. Листинг алгоритма5

**6. Библиографический список10**

1. **Общее описание алгоритма** 
   1. **Назначение алгоритма**Разработанный алгоритм служит для создания модели сверточной нейронной сети, способной дать ответ о наличии присутствия дрона на звуковой дорожке.
   2. **Теоретическое описание алгоритма**

Сверточная нейронная сеть предназначена для определения принадлежности исследуемого объекта к какому-то определенному классу объектов благодаря вычленению признаков из объекта и оценки их схожести с признаками объектов тех классов, к которым этот объект будет отнесем. В нашем случае будет существовать только два класса принадлежности: есть дрон на звуковой дорожки или нет. Задача состоит в том, чтобы подобрать такие гиперпараметры модели нейронной сети и выбрать такие признаки из набора тестовых данных, которые будут наиболее существенными для исследуемого объекта и позволят максимально точно определить принадлежность объекта к определенному классу.

1. **Реализация алгоритма**

**2.1 Описание компонентов алгоритма**

**Используемые библиотеки:**

* Librosa – библиотека для работы со звуком.
* Pandas – мощный пакет для анализа данных.
* Matplotlib – диблиотека на языке программирования Python для визуализации данных двумерной (2D) графикой.
* CVS – данный модуль даёт программисту возможность выполнять структурный анализ CVS файлов.
* SKlearn– библиотека машинного обучения на языке программирования Python.
* Keras – открытая нейросетевая библиотека.

**Входные данные:**Для обучения и тестирования нейронной сети использовался датасет из открытого репозитория:   
<https://github.com/saraalemadi/DroneAudioDataset/tree/master/Binary_Drone_Audio>

Датасет содержит 2 класса:   
1. Есть дрон (1332 звуковые дорожки)  
2. Нет дрона (1403 звуковые дорожки)

**2.2 Детали реализации алгоритма**

**2.2.1.** На первом шаге необходимо для каждой звуковой дорожки из датасета создать соответствующие спектрограммы для упрощения извлечения сущностей. Спектрограмма – это визуальное представление спектра частот звуковых сигналов, изменяющихся со временем.

Мы будем извлекать такие сущности как:

* Спектральный центроид - указывает, где расположен "центр масс" звука, и рассчитывается как средневзвешенное значение всех частот.
* Частота перехода через нуль - это частота изменения знака сигнала, т. е. частота, с которой сигнал меняется с положительного на отрицательный и обратно.
* Спектральная ширина - интервал частот или интервал длин волн, характеризующий спектральные линии в излучениях квантовых систем.
* Спектральный спад частоты - это мера формы сигнала, представляющая собой частоту, ниже которой лежит определенный процент от общей спектральной энергии.
* Частота цветности - это интересное и мощное представление для звука, при котором весь спектр проецируется на 12 контейнеров, представляющих 12 различных полутонов музыкальной октавы.

Извлечение сущностей из аудиодорожек производилось с помощью функций библиотеки Librosa.

**2.2.2** На следующем шаге извлеченные сущности будут записаны в CVS файл с помощью методов модуля CVS для их дальнейшего анализа.

**2.2.3** Следующим шагом будет предобработка (масштабирование) данных обучения из CVS файла – это необходимо для обучения модели. Масштабирование данных производилось с помощью методов библиотеки SKlearn.

**2.2.4** Следующий шаг – разбиение данных на обучающую и тестовую выборку, выло выбрано соотношение 80% обучающих данных, 20% тестовых. Здесь также использовалась библиотека SKlearn.

**2.2.5** На следующем шаге осуществляется построение модели: настройка слоёв нейронной сети, настройка конфигурации обучения. На данном шаге подбор параметров производится методом подбора, для того, чтобы найти такие параметры, при которых модель будет максимально точно делать предсказания. Здесь для построения модели используется библиотека Keras.

**2.2.6** На следующем шаге производится обучение модели на 80% всех данных так же с помощью методов библиотеки Keras.

**3. Тестирование**

**3.1 Методы проведения тестирования**

Тестирование проводилось в два этапа:

1. Модель оценивается на 20% данных из датасета.
2. Модель проверялась на записях дронов (20 шт.), которые были записаны своими силами.

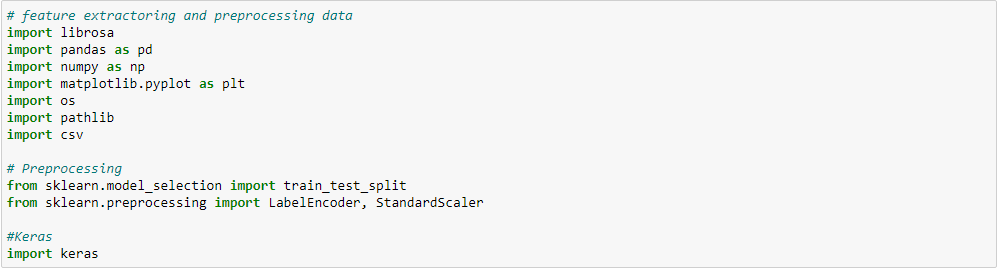
**3.2 Результаты тестирования**

По результатам 1 этапа тестирования на тестовых данных модель показала точность 86%.  
По результатам 2 этапа тестирования на собственных записях модель показала точность ~ 90%. Количество записей: 20 шт.

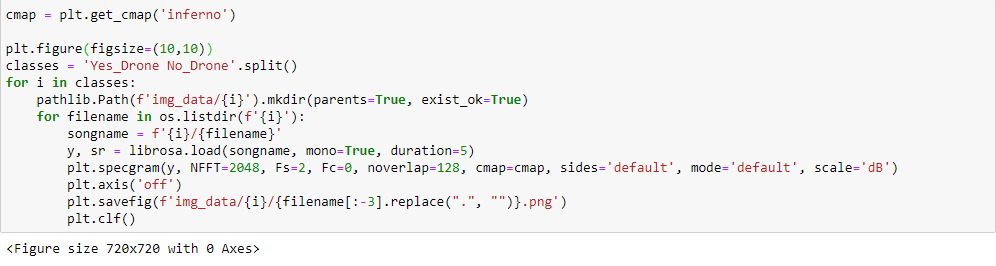
**4. Интеграция с другими компонентами системы.** Предполагается интеграция модели нейронной сети в клиент серверную часть. Клиент часть будет отправлять данные на сервер, где они будут обрабатываться нейронной сетью и давать предсказание о наличии или отсутствии дрона.

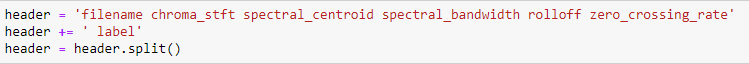
**5. Листинг алгоритма.**

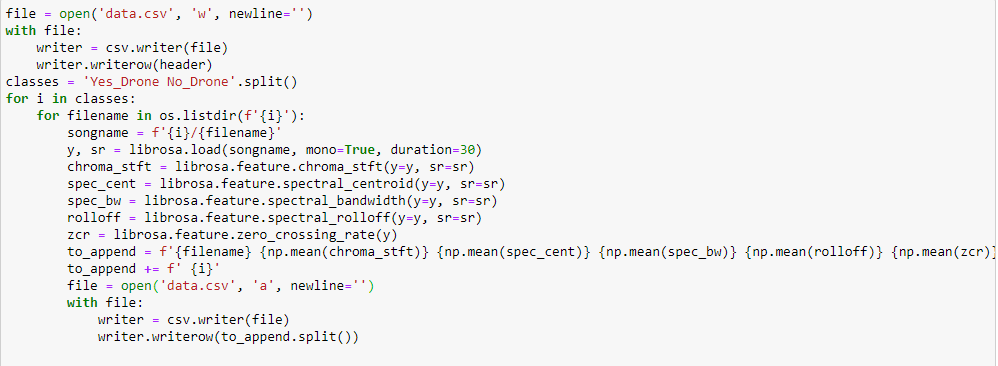
**Импорт библиотек:**



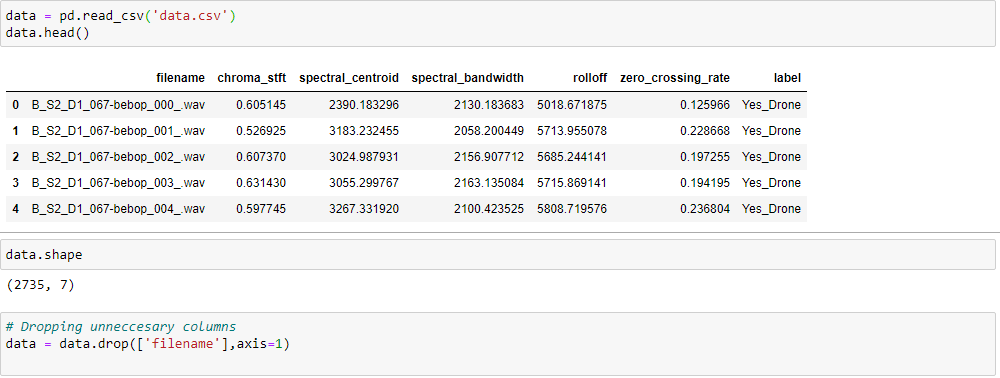
**Извлечение спектрограмм для каждой звуковой дорожки:**



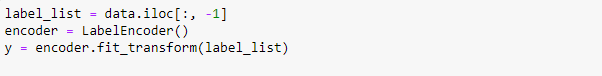
**Извлечение сущностей из спектрограмм и запись данных в CVS файл**  




**Чтение данных с помощью Pandas:**



**Кодирование ярлыков принадлежности к определенному классу**



**Предобработка (масштабирование) данных для обучения**

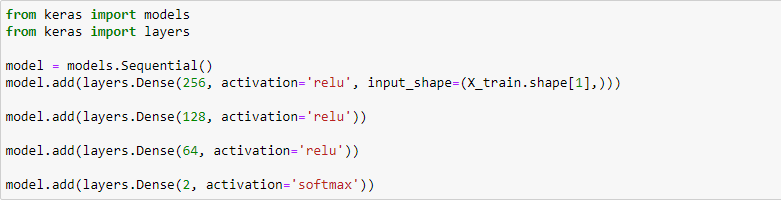


**Разделение данных на обучающий и тестовый набор**



**Построение модели нейронной сети**

Настройка слоёв сети:



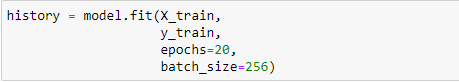
Настройка конфигурации обучения:

Прежде чем модель будет готова для обучения, нам нужно указать еще несколько параметров. Они добавляются на шаге compile модели:

* Функция потерь (Loss function) — измеряет точность модели во время обучения. Мы хотим минимизировать эту функцию чтоб "направить" модель в верном направлении.
* Оптимизатор (Optimizer) — показывает каким образом обновляется модель на основе входных данных и функции потерь.
* Метрики (Metrics) — используются для мониторинга тренировки и тестирования модели. Наш пример использует метрику accuracy равную доле правильно классифицированных аудиодорожек.



**Обучение модели**



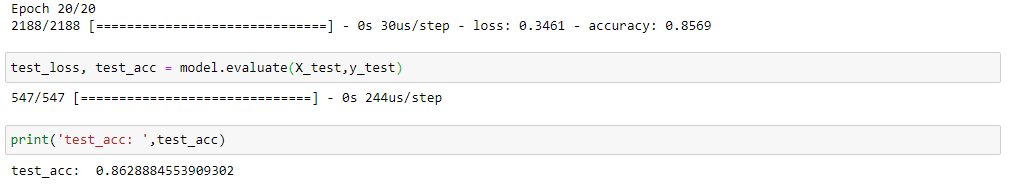
**Сохранение обученной модели и предобработчика данных**



В дальнейшем модель может быть использована для генерации предсказаний.

**Тестирование модели**

1 этап тестирования:

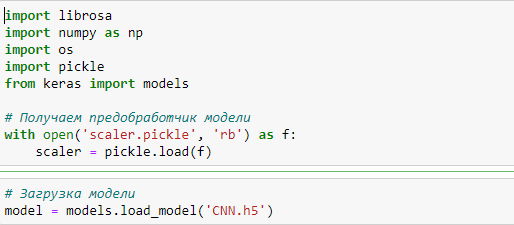


Точность на тестовых данных (20%) из датасета равна 86%.

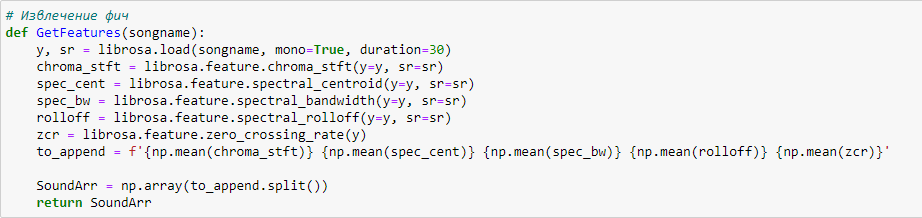
2 этап тестирования:

Используя сохранённую модель обученной нейронной сети можно генерировать предсказания для входных аудиодорожек.

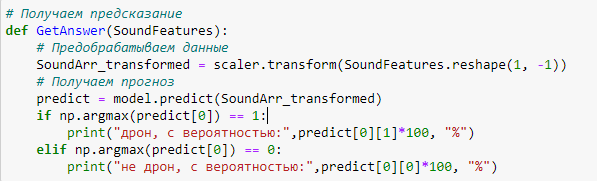
Подгружаем модель нейронной сети (CNN.h5) и предобработчик данных (scaler.pickle), который создавался и использовался в процессе обучения модели. Это необходимо, т.к. для получения предсказания для новых данных мы должны извлекать из них те же самые сущности и масштабировать эти данные таким же образом, как и при обучении модели.



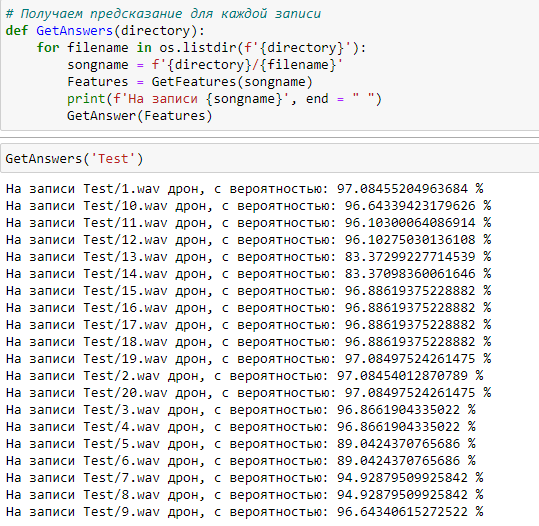
Создаем функцию для извлечения сущностей из аудиодорожки. Эта функция принимает в параметре название аудиодорожки, а возвращает массив с извлеченными сущностями:



Создаем функцию для получения предсказаний для аудиодорожки, которая принимает в параметре массив сущностей из интересующей аудиодорожки. Данная функция предобрабатывает данные в массиве с помощью объекта scaler, после чего с помощью метода model.predict(array) загруженной модели, принимающего в параметре массив с обработанными данными, генерируется предсказание. Т.к. у нас существует 2 класса (есть дрон, нет дрона), по принадлежности к какому-либо классу можем дать ответ о наличии или отсутствии дрона на звуковой дорожке.



Далее создаем функцию генерации всех ответов для интересующих нас аудиодорожек и получаем для них предсказания:



На каждой из проверяемых аудиодорожек присутствовали звуки дрона.  
По результату работы функции можно увидеть, что модель нейронной сети дала правильный ответ с результатом не ниже 89% точности.

**Библиографический список**

[**https://www.tensorflow.org/tutorials/keras/classification?hl=ru**](https://www.tensorflow.org/tutorials/keras/classification?hl=ru)

[**https://gist.github.com/parulnith/7f8c174e6ac099e86f0495d3d9a4c01e#file-music\_genre\_classification-ipynb**](https://gist.github.com/parulnith/7f8c174e6ac099e86f0495d3d9a4c01e%23file-music_genre_classification-ipynb)