МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

Механико-математический факультет

Кафедра математической теории интеллектуальных систем

Курсовая работа

ОСНОВНЫЕ ЗВУКОВЫЕ ХАРАКТЕРИСТИКИ РАСПОЗНАВАНИЕ РЕЧЕВЫХ ЭМОЦИЙ BASIC SOUND FEATURES RECOGNITION OF SPEECH EMOTIONS

Подготовил:
студент 333 группы
Зеленин Герман Евгеньевич

Научный руководитель:
доцент каф. математической теории интеллектуальных систем
Миронов Андрей Михайлович

Москва, 2022г.

1 Аннотация

В данной курсовой работе пойдет речь об использовании свёрточных нейронных сетей для классификации звуковых записей голоса людей. Будут описаны следующие звуковые характеристики: MFCC (Mel-кепстральные коэффициенты) и спектрограммы. Описан принцип работы быстрого преобразования Фурье и построение Mel-спектрограмм. Создана сверточная нейронная сеть, способная с достаточно хорошо (с 72 процентной точностью) для данного уровня классифицировать настроение говорящего человека.

2 Введение

Распознавание речевых эмоций (Speech Emotion Recognition, SER) - это попытка распознать эмоции человека по его речи, которая основана на том факте, что голос часто отражает основные эмоции через тон голоса и его высоту.

2.1 Постановка задачи

Главной задачей этой курсовой работы является изучение основных характеристик звука и их связи между собой, выявление взаимосвязи между этими характеристиками и настроением человека с помощью сверточных нейронных сетей.

2.2 Мотивация исследования

Распознование эмоций - развивающийся раздел машинного обучения. Потребность в этом растет повседневно. Где это может применяться? Например, SER используется во всех основных голосовых ассистентах при создании ответа на запрос пользователя. Для бизнеса очень важно собирать данные о состоянии человека, который обратился к автоответчику или чат-боту. Это позволит выявить упущения в какой-либо части своего производственного процесса и изменить его в лучшую сторону. Также это может использоваться в различных местах, где важно, чтобы человек был в нормальном психическом состоянии. Например, автомобильные системы могли бы не позволять человеку выехать на дорогу, если его речь агрессивна (как следствие раздражительности и пошатнувшегося психического здоровья). Понятно, что учитывать во всех этих примерах только речь нельзя, это лишь один из способов выявления психических отклонений у человека.

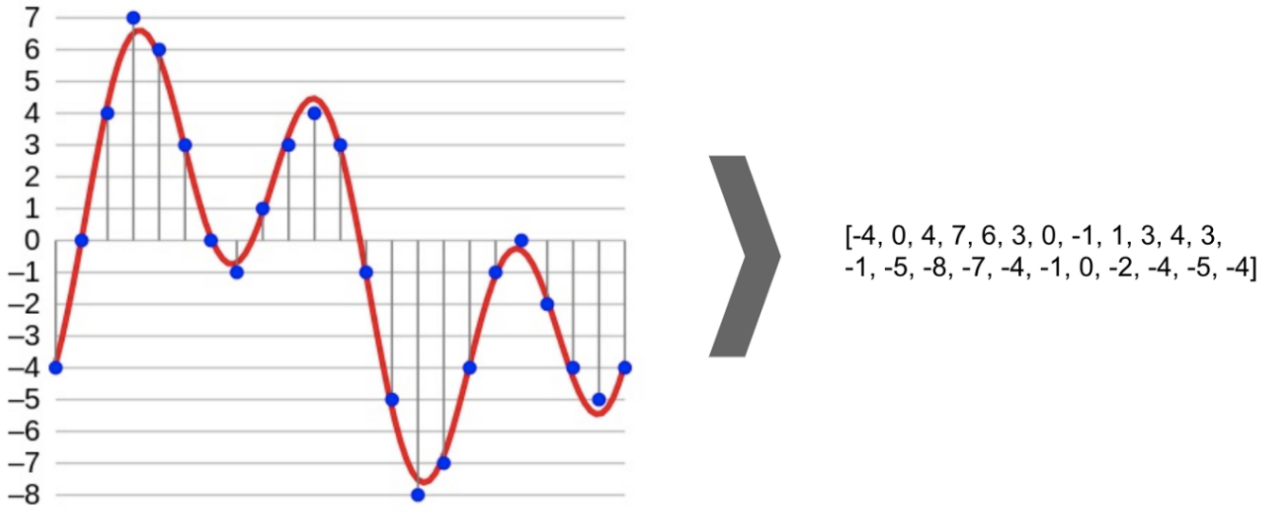
3 Основные понятия

Звук - это физическое явление, представляющее собой распространение в виде упругих волн механических колебаний в твердой, жидкой или газообразной среде. Как и любая волна, звук характеризуется **амплитудой** и **частотой**. **Амплитуда** характеризует громкость звука, а **частота** определяет тон и высоту.

Звуковые волны оцифровываются путем их дискретизации с интервалами, известными как **частота дискретизации**. Обычно используется частота дискретизации равная 44,1 кГц. Это означает, что на секундном отрезке отбираются значения амплитуды 44100 раз через равные промежутки времени.

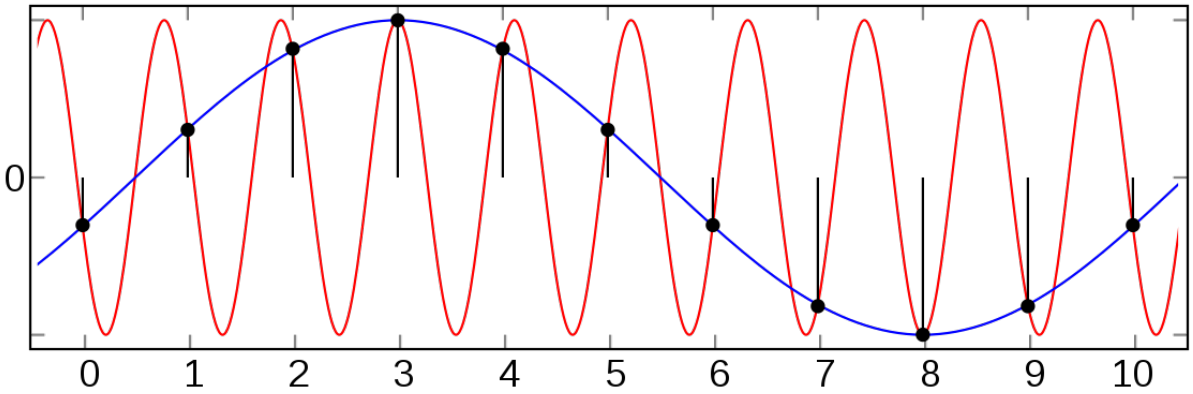
Введем понятие **битовой глубины** (bit-depth). Эта величина обозначает то, насколько детализирована выборка по амплитуде. То есть, если bit-depth = 16, то амплитуда может принять 65536 значений.

Процесс дискретизации можно показать следующим образом:



Рассмотрим еще одно понятие, которое относится к периодическим функциям. **Частота сигнала** (англ. frequency) - это число периодов функции за 1 секунду.

Однако существует проблема: сигнал определенной частоты на большом шаге дискретизации имеет в результате дискретизации частоту, меньшей исходной. Наиболее наглядно данный эффект, называемый алиасингом, можно продемонстрировать данной иллюстрацией:

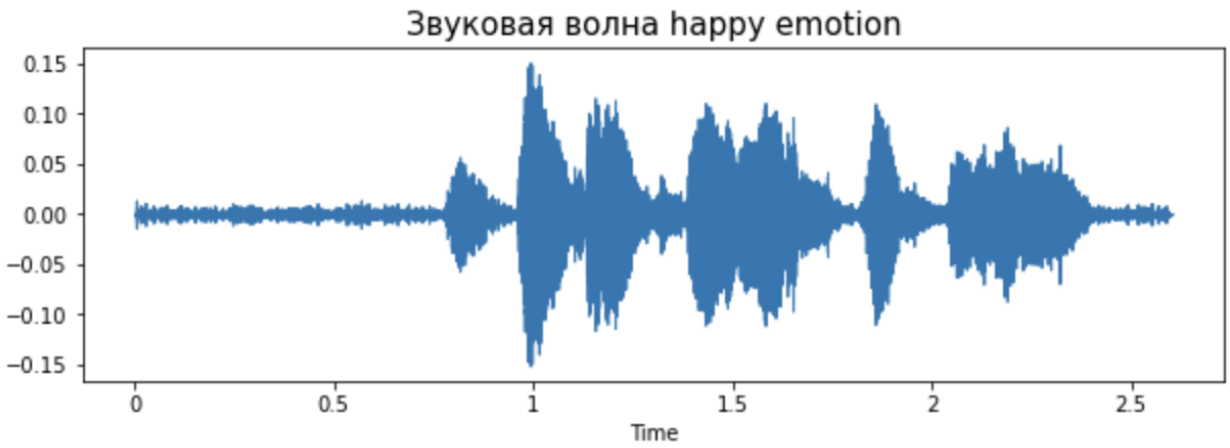


Здесь видно, что по полученной выборке амплитуд получается функция (синим цветом), у которой частота куда ниже, чем у красной волны. Для решения данной теоремы существует теорема Котельникова.

Теорема Котельникова (Nyquist theorem): частота дискретизации должна быть в 2 или более раз выше, чем масимальная частота, которую мы хотим записать без алиасинга.

4 Быстрое преобразование Фурье

Как уже было сказано, любой звуковой сигнал - это дискретный массив, содержащий значения амплитуды через равные, очень маленькие промежутки времени. Мы можем построить сигнал какого-то аудиофайла, например:



Здесь, по горизонтальной оси - время с начала записи, по вертикальной - амплитуда. Было бы удобно перейти из временной области в частотную. Любой звук - это наложение разных синусоид и косинусоид со своей частотой и амплитудой. Было бы хорошо разложить сигнал на **спектр**. На помощь приходит дискретное преобразование Фурье [1,2]. Оно помогает разложить входной сигнал на отдельные частоты и соответствующие им амплитуды.

Пусть N - количество значений сигнала, измеренных за период времени T, x_n, где $n = 0, ..., N-1$ - измеренные значения сигнала в дискретные моменты времени, проиндексированные n. Тогда комплексные амплитуды синусоидальных сигналов, слагающих исходный сигнал выражаются следующим образом:

$$X_k = \sum_{n=0}^{N-1} x_n(\cos(2\pi kn/N) + i\sin(2\pi kn/N))$$

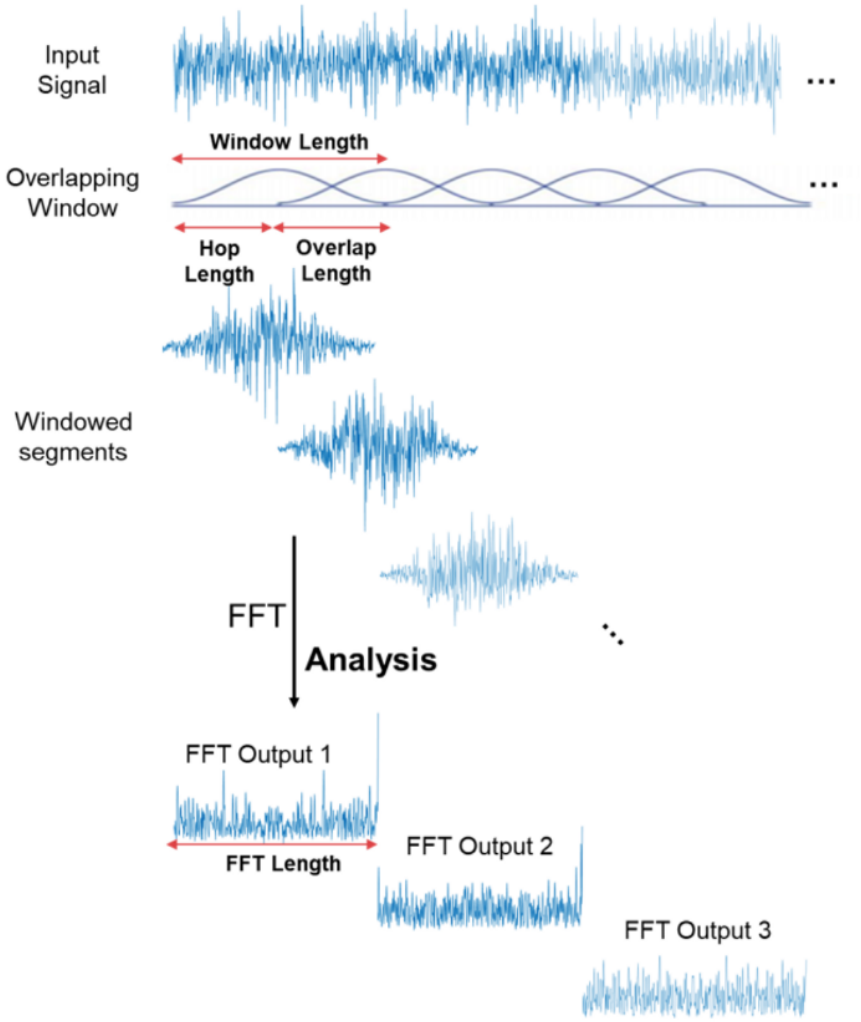
Поскольку амплитуды комплексные, то по ним можно вычислить одновременно и амплитуду, и фазу. Тогда вещественная амплитуда k-ого синусоидального сигнала выражается как $\frac{|X_k|}{N}$, $\arg(X_k)$ - фаза k-ого синусоидального сигнала, k - собственно индекс частоты $\frac{k}{T}$, T - период времени на котором брался сигнал.

Дискретное преобразование Фурье (ДПФ) для своей реализации требует выполнения N^2 умножений комплексных чисел. ДПФ может быть сильно упрощено, если использовать свойства симметрии и периодичности коэффициентов. Результатом переработки выражений для ДПФ является быстрое преобразование Фурье (БПФ), которое требует только $\frac{N\log_2 N}{2}$. Вычислительная эффективность БПФ по сравнению с ДПФ становится весьма существенной, когда количество точек БПФ увеличивается до нескольких тысяч. Очевидно, что БПФ вычисляет все компоненты выходного спектра. Если необходимо рассчитать только несколько точек спектра, ДПФ может оказаться более эффективным. Вычисление одного выходного отсчета спектра с использованием ДПФ требует только N умножений с комплексными числами. [2]

5 Спектрограмма

Использование быстрого преобразования Фурье на аудиофайлах длиной 2-3 секунды не несет полезной информации. Поэтому предлагается разбиваться звук на много маленьких последовательных кусочков и уже их преобразовывать БПФ. Рассмотрим как мы это будем делать.

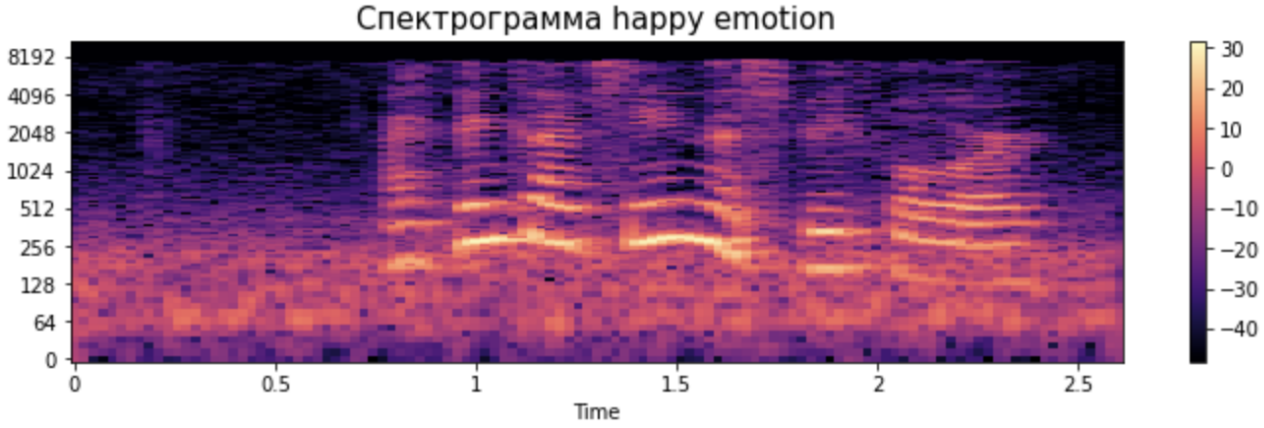
Пусть L - длина сигнала (на 1 секунде $L = 44100$). Мы будем производить преобразование Фурье на маленьких участках звукового сигнала. Обозначим длину такого участка как W, будем называть его "окном". Обычно $W = 1024$ или 2048. Данное окно будет постоянно сдвигаться на величину H (Hop). Обычно его берут как четверть ширины окна. Следующая иллюстрация наглядно показывает как строится спектрограмма:



Человеческое ухо более чувствительно к изменениям звука на низких частотах, чем на высоких. То есть если частота звука изменится с 100 Гц до 120 Гц, то человек это заметит, но изменение с 10 000 Гц до 10 020 Гц - не заметит. Поэтому вводится новая единица измерения звука - **мел**. Она основана на психофизиологическом восприятии звука человеком. $1mel = 1127.01048 \ln(1+\frac{freq}{700})$.

Помимо этого часто используют еще одну единицу измерения - **бел**. Это дольная единица, применимая к амплитуде. Увеличение на 1дБ означает увеличение в $10^{0.1}$. $D_p = 10 \log(\frac{P_2}{P_1})$.

Итак, спектрограмма - двумерная диаграмма: на горизонтальной оси представлено время, по вертикальной оси — частота; третье измерение с указанием амплитуды на определенной частоте в конкретный момент времени представлено интенсивностью или цветом каждой точки изображения. Покажем как это выглядит:



Кроме того мы можем строить и мелспектрограмму заменяя частотную ось на мел-ось по формуле, описанной выше.

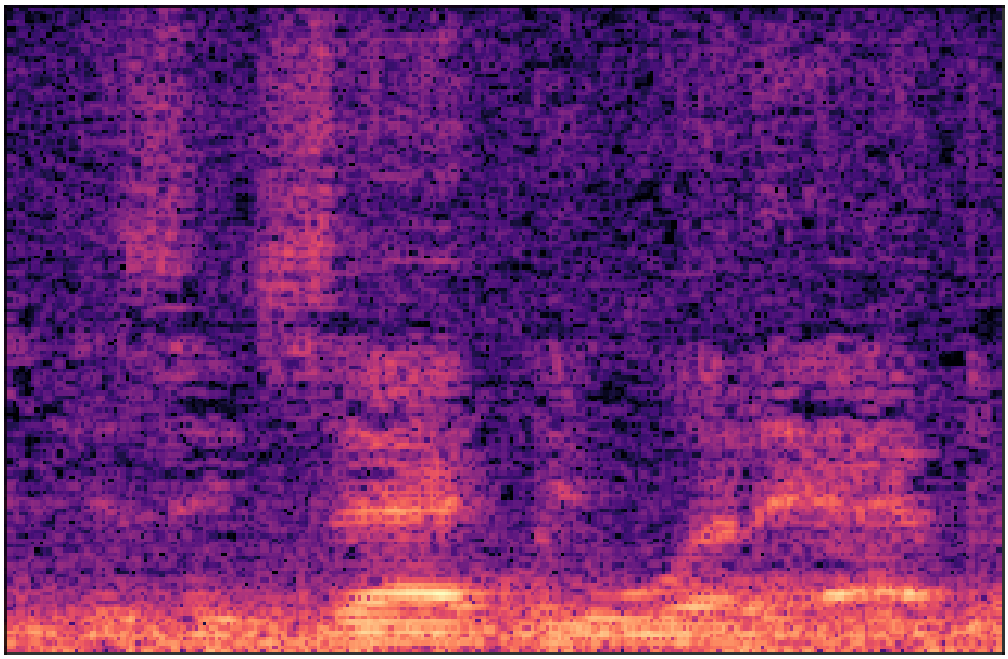
6 Классификация с помощью нейронных сетей по спектрограмме

Первый результат этой курсовой работы показывает, что ограничиваться лишь анализом спектрограммы нельзя. Однако такой способ дает неплохой результат для бинарной классификации (хорошее / плохое настроение человека). Для данного "опыта"использовался набор данных записей голосов актеров CREMA-D. Он содержит 7442 коротких предложений, которые произносятся мужчинами и женщинами разного возраста. Обработка происходила следующим образом:

1. **Разметка данных**
   Данные в датасете представлены как аудиофайлы расширения ".wav". В названии каждого аудиофайла фигурирует та эмоция, которая произнесена человеком (Sad - Печаль, Andry - злость, Disgust - отвращение, Fear - страх, Happy - радость, Neutral - нейтральный). Однако по ходу работу окажется, что классифицировать по каждой из эмоций по спектрограмме - нерационально и дает плохой результат, поэтому было решено сделать бинарную классификацию (положительная (Neutral, Happy) / Отрицательные (остальные)). Минус данного датасета - отсуствие разметки по полу говорящего. Классификация голоса мужчины и женщины без различия вносит свою ошибку, как правило тоны голоса различаются.

2. **Построение спектрограмм**
   Для каждой записи строилась спектрограмма. Амплитуды на каждой записи переводились в шкалу Db. Исследуя весь датасет было выяснено, что минимальное значение амплитуды в Db было равно -100, максимальное 55. Было решено, что можно все значения амплитуды можно "снести т.е.

-100 в новой нумерации становится 0, а 155 становится 55. Это возможно, т.к. все звуки были записаны в одинаковых условиях, поэтому исследуется разница амплитуд, а не ее значение. Далее, из каждой спектрограммы берется середина (примерно там, где человек говорит). Размеры полученных спектрограмм 288 на 432 пикселя. Пример такой спектрограммы:



3. **Разложение спектрограммы в RGB**
   Каждая спектрограмма разлагалась в тензор размерности (288 * 432 * 3) - по цветам RGB. На каждом месте лежало значение от 0 до 255. Прежде чем подавать данный тензор в нейронную сеть его нужно нормировать.

4. **Построение нейронной сети**
   Итак, имея тензоры спектрограмм размерности (288 * 432 * 3) можем использовать их для обучения нейронной сети. Весь датасет делится в отношении 80/20 (обучающая выборка/тестирующая выборка). Структура нейронной сети показана на последующей картинке. Код нейронной сети будет представлен в **Листинг 1**.

```
_________________________________________________
Layer (type)                 Output Shape
=================================================
conv2d_28 (Conv2D)           (None, 288, 432, 32)

max_pooling2d_28 (MaxPoolin  (None, 32, 48, 32)
g2D)

conv2d_29 (Conv2D)           (None, 32, 48, 64)

max_pooling2d_29 (MaxPoolin  (None, 16, 24, 64)
g2D)

conv2d_30 (Conv2D)           (None, 16, 24, 128)

max_pooling2d_30 (MaxPoolin  (None, 8, 12, 128)
g2D)

conv2d_31 (Conv2D)           (None, 8, 12, 256)

max_pooling2d_31 (MaxPoolin  (None, 4, 6, 256)
g2D)

max_pooling2d_32 (MaxPoolin  (None, 2, 3, 256)
g2D)

flatten_7 (Flatten)          (None, 1536)

dense_18 (Dense)             (None, 129)

dense_19 (Dense)             (None, 2)
```

5. **Результаты**

   В результате обучения и тестирования сети был получен результат точности бинарной классификации: accuracy = 0.7282. Это неплохой результат, но не тот, что ожидают от нейронной сети. Это объясняется многими факторами.

   Первое - плохая разметка данных. Как уже говорилось выше, данные не были размечены по полу человека. Фактически, голоса мужчин и женщин были приравнены друг к другу, хотя из жизненного опыта известно, что это не так. Исследования показывают, что голосовые связки у мужчин обычно более массивные и более длинные, чем у женщин, поэтому частота основного тона мужского голоса в норме находится в диапазоне от 80 Гц до 240 Гц, а женского - от 140 Гц до 500 Гц.

   Второе замечание - индивидуальность. У каждого человека свои харак-

теристики при той или иной эмоции. При злости у каждого человека повышается голос сильнее, чем у другого.

Третье - индивидуальность при трактовке эмоций. Разметка данных происходила лишь от того, как воспринимает запись один человек.

Четветое - несовершенность нейронной сети. Однако как показывают другие исследования [3], используя такой же метод классификации спектрограмм с помощью нейронной сети AlexNet был достигнут результат лишь около 0.78, что недалеко от моего результата.

Из всего этого можно сделать вывод, что классификация по спектрограмме - очень затратная задача, которая не оправдывается. А значит, нужно искать другие характеристики звука, которые помогли бы в классификации.

7 Мел-частотные кепстральные коэффициенты (MFFC)

Построение спектрограммы и ее анализ - тяжелый способ анализа звукового сигнала. Изучение проблемы привело к тому, чтобы найти другие коэффициенты, которые быстрее и удобнее позволяли анализировать сигнал, дающие лучший результат.

Мел-кепстральные коэффициенты были введены S. Davis и P. Mermelstein [4]. До этого основными характеристиками для распознавания речи были линейные коэффициенты предсказания и линейные кепстральные коэффициенты предсказания. Рассмотрим как строются данные коэффициенты:

1. **Разбиение сигнала:**

   Необходимо разбить изначальный сигнал на фреймы. Размер фрейма выбирается от 20 до 40 мс (или как мы делали раньше: 512-2048 дискретных значений сигнала). Считается, что на таком маленьком промежутке сигнал не изменяется. В итоге, речевой сигнал записывается как $x(n), 0 \leq n < N$, где N - размер фрейма, $x_j(n)$ - j-ый фрейм. Следующие шаги применяются для каждого отдельного фрейма.

2. **Функция Хемминга:**

   Речевой сигнал конечен и не является периодическим, поэтому на из-за разрывов на его концах происходит эффект "утечки"при преобразовании

Фурье. Для того, чтобы снизить его влияние на результат, каждый кадр умножается на оконную функцию Хемминга:

$$w(n) = 0.54 - 0.46\cos(\frac{2\pi n}{N-1}), 0 \leq n \leq N-1$$

К получившемуся результату применяем дискретное преобразование Фурье:

$$X_j(k) = \sum_{n=0}^{N-1} x_j(n)w(n)e^{-\frac{2\pi i}{N}kn}, 0 \leq k < N$$

где j-ый номер фрейма.

3. **Периодограмма:**

   Вычисляем периодограмму для каждого фрейма по следующей формуле:

$$P_j(k) = \frac{|X_j(k)|^2}{N}$$

4. **Мел-фильтры:**

   Каждый фильтр моделируется с помощью следующей функции:

$$H_m(k) = \begin{cases} 0, & k < f(m-1) \\ \frac{k-f(m-1)}{f(m)-f(m-1)}, & f(m-1) \leq k < f(m) \\ \frac{f(m+1)-k}{f(m+1)-f(m)}, & f(m) \leq k \leq f(m+1) \\ 0, & k > f(m+1) \end{cases},$$

   где m - число фильтров, которое мы хотим получить. Зная число фильтров и диапозон интересующих нас частот, функции $f(x)$ можно найти, используя формулы преобразования в мел частоту и обратно.

5. **MFCC:**

   Полученные энергии логарифмируются. Это также мотивируется человеческим слухом: мы не слышим громкость в линейном масштабе. Обычно, чтобы удвоить воспринимаемую громкость звука, нам нужно затратить в 8 раз больше энергии. Это означает, что большие колебания энергии могут звучать не так уж и по-другому, если звук с самого начала громкий. Эта операция сжатия делает наши функции более близкими к тому, что на самом деле слышат люди. Мы получаем некоторый набор коэффици-

ентов, которые еще не являются MFCC:

$$S_j(m) = \ln \sum_{k=0}^{N-1} P_j(k) H_m(k), 0 \leq m < M$$

Далее, используя дискретное косинусное преобразование, получим мел-кепстральные коэффициенты:

$$c_j(n) = \sum_{m=0}^{M-1} S_j(m) cos(\pi n(m + 1/2)/M), 0 \leq n < M$$

8 Классификация с помощью нейронных сетей по MFCC

В данной задаче попробуем отойти от простой бинарной классификации равных классов (Negative / Positive) и перейдем к сложной классификации по 6 основным классам.

В ходе работы с данными использовались следующие библиотеки:

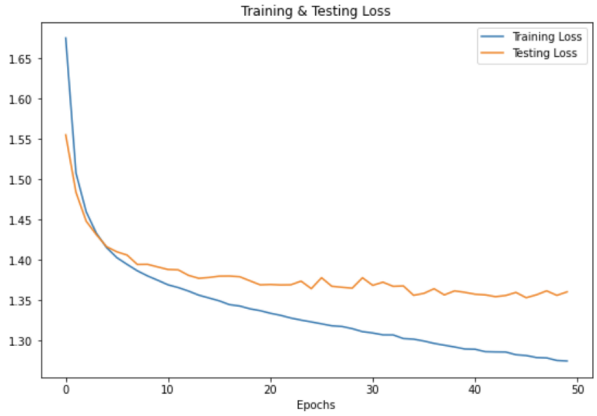
1. **Pandas:** для загрузки данных и работы с ними;
2. **NumPy:** для работы с массивами;
3. **Librosa:** для загрузки аудиофайлов и работы с ними (Librosa позволяет строить спектрограммы и вычислять коэффициенты MFCC);
4. **Keras:** для построения нейронной сети;

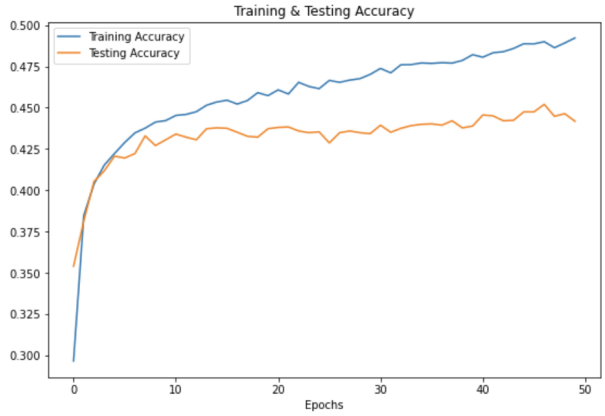
Учтем некоторые ошибки при первой обработке данных. Единственное что мы можем сделать - увеличим число данных. В изначальном датасете мы имеем 7442 аудиозаписи. Добавим к каждой записи "шум". Таким образом мы получим еще больше записей, которая имеет нормальную и "шумную"версию. Это позволит обучить сеть на работу с зашумленными данными. Как зашумляется аудиозапись показано в **Листинге 2**.

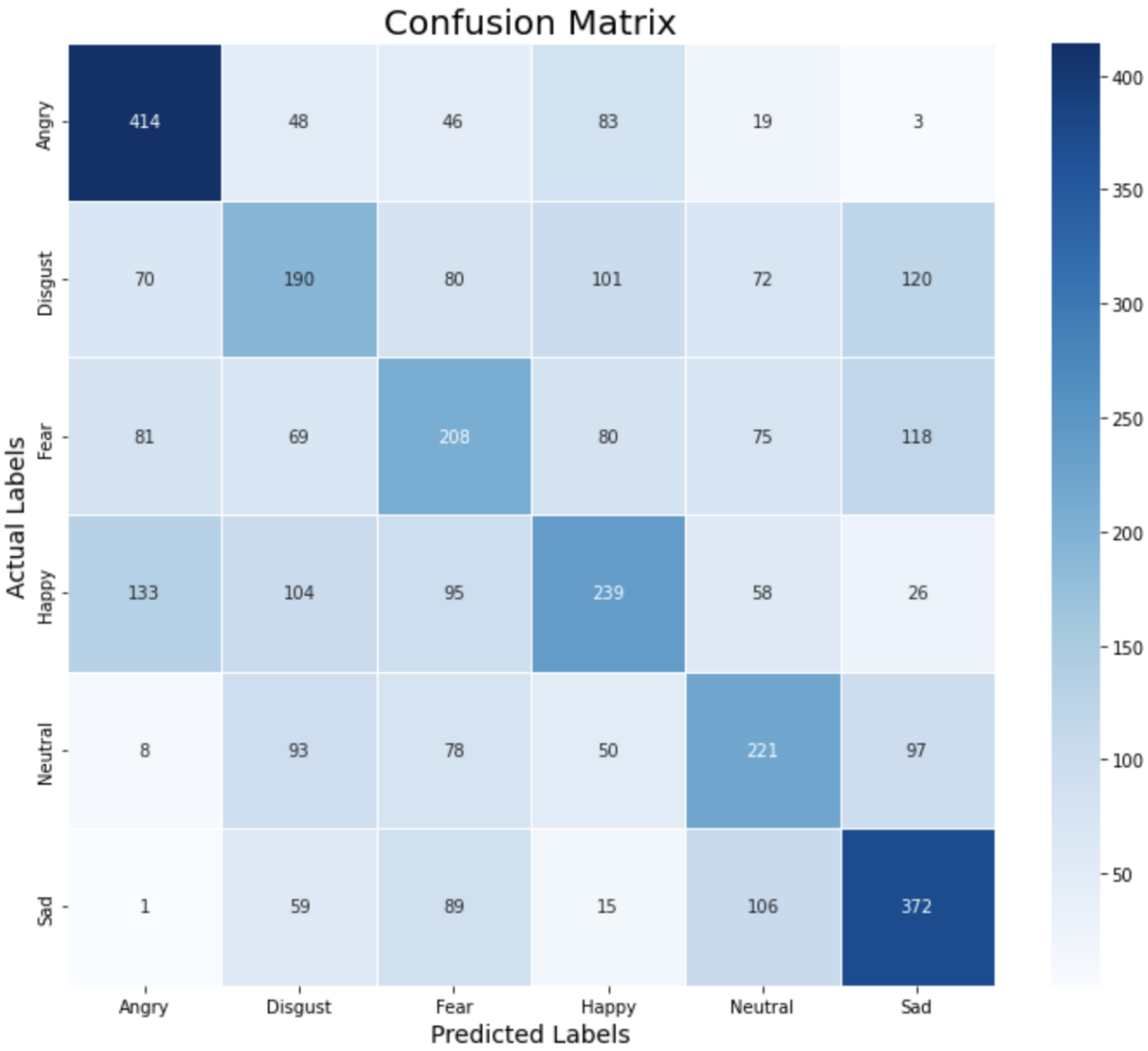
Подсчитаем 20 мел-частотных кепстральных коэффициентов для каждой записи (среднее по каждому отрывку из аудиозаписи). Попытаемся классифицировать с помощью нейронной сети по MFCC. Код нейронной сети указаны в **Листинге 3**.

Результаты по обучению представлены в графике:

```
117/117 [==============================] - 0s 969us/step - loss: 1.3605 - accuracy: 0.4418
Accuracy of our model on test data :  44.18167173862457 %
```







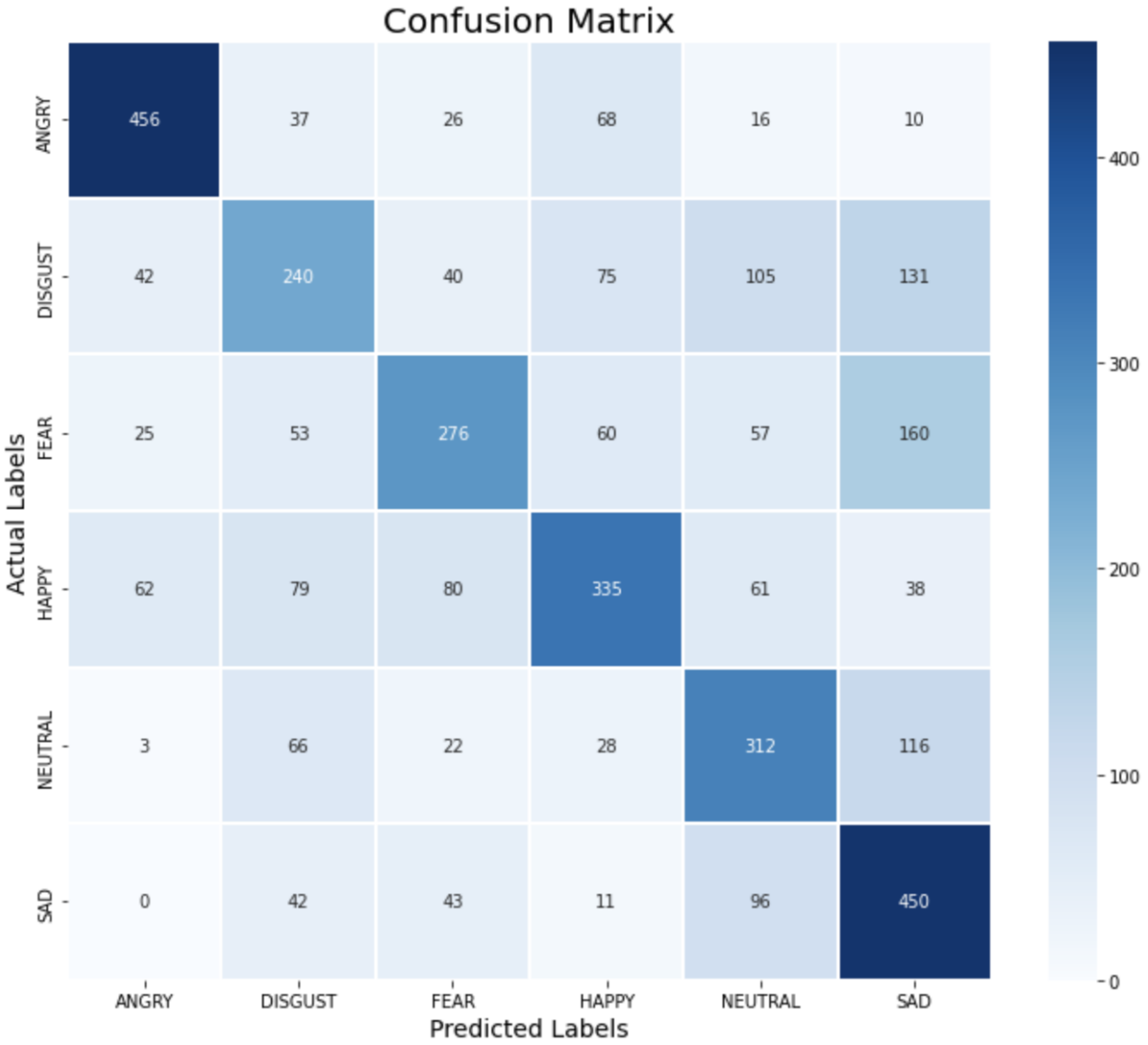
Вывод:

Как мы видим, достаточно неплохо выделяется диагональ на нашей матрице, модель показывает плохой результат на парах Happy - Angry, Disgust - Sad,

Fear - Sad, Disgust - Happy. Почему так происходит. Основная причина в том, что те величины, которые мы используем (MFCC) не могут передать отличительные черты той или иной эмоции. Как говорилось выше, свой вклад в ошибку вносит плохая разметка данных и человеческий фактор при ее разметке. Попробуем объединить воедино два метода для предсказания. Будем брать средние по мелспетрограмме и средние по MFCC.

9 Классификация с помощью нейронных сетей по MFCC и спектрограмме:

В третьем случае будем использовать средние мел-частотных кепстральных коэффициентов и средние по спектрограмме. Результат гораздо улучшился. Матрица предсказаний представлена ниже. В **Листинг 4** представлена нейронная сеть для классификации в данном случае.



10 Общий вывод:

В данной курсовой работе мною были освоены основы построения звуковых характеристик: MFCC и Спектрограммы. Были применены нейронные сети для попытки классификации 6 различных эмоций в речи человека. Результат достаточно неплох, поскольку тренировочный dataset плохо размечен, а также, как показывает практика, мел-частотных кепстральных коэффициентов недостаточно для хорошей классификации. В данном направлении будет продолжена работа.

Листинг 1

```
model = tf.keras.Sequential([
    tf.keras.layers.Conv2D(32, (3,3), padding='same',
                activation=tf.nn.relu, input_shape=(288, 432, 3)),
    tf.keras.layers.MaxPooling2D((2, 2), strides=2),
    tf.keras.layers.Conv2D(64, (3,3), padding='same',
                activation=tf.nn.relu),
    tf.keras.layers.MaxPooling2D((2, 2), strides=2),
    tf.keras.layers.Conv2D(128, (3,3), padding='same',
                activation=tf.nn.relu),
    tf.keras.layers.MaxPooling2D((2, 2), strides=2),
    tf.keras.layers.Conv2D(256, (3,3), padding='same',
                activation=tf.nn.relu),
    tf.keras.layers.MaxPooling2D((2, 2), strides=2),
    tf.keras.layers.Flatten(),
    tf.keras.layers.Dense(176, activation=tf.nn.relu),
    tf.keras.layers.Dense(32, activation=tf.nn.relu),
    tf.keras.layers.Dense(6, activation=tf.nn.softmax)
])

model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy', metrics=['accuracy'])

his = model.fit(X_trn, y_trn, batch_size = 16, epochs = 15,
                                        validation_split = 0.2)
```

Листинг 2

```
def noise(data):
    noise_amp = 0.035*np.random.uniform()*np.amax(data)
    data = data + noise_amp*np.random.normal(size=data.shape[0])
    return data

def extract_features(data, sample_rate):
    result = np.array([])
    mfcc = np.mean(librosa.feature.mfcc(y=data, sr=sample_rate).T,
                                                        axis=0)
    return result = np.hstack((result, mfcc))

def get_features(path):
    data, sample_rate = librosa.load(path, duration=2.5,
```

```
                                                    offset=0.6)
    res1 = extract_features(data, sample_rate)
    result = np.array(res1)
    noise_data = noise(data)
    res2 = extract_features(noise_data, sample_rate)
    result = np.vstack((result, res2))
    return result
```

Листинг 3

```
model=Sequential()
model.add(Flatten(input_shape=(x_train.shape[1], 1)))
model.add(Dense(units=32, activation='relu'))
model.add(Dense(units=16, activation='relu'))
model.add(Dense(units=6, activation='softmax'))
model.compile(optimizer = 'adam' ,
loss = 'categorical_crossentropy' , metrics = ['accuracy'])
```

Листинг 4

```
model.add(Flatten(input_shape=(x_train.shape[1], 1)))
model.add(Dense(units=128, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(units=64, activation='relu'))
model.add(Dropout(0.2))
model.add(Dense(units=6, activation='softmax'))
model.compile(optimizer = 'adam' ,
loss = 'categorical_crossentropy' , metrics = ['accuracy'])
```

Список литературы

[1] А.Б. Сергиенко. Цифровая обработка сигналов. 3-е изд / СПб.: БХВ-Петербург, 2011.

[2] В.П. Кандидов, Дискретное преобразование Фурье. / Москва: физический факультет МГУ, 2019.

[3] Margaret Lech, Real-Time Speech Emotion Recognition Using a Pre-trained Image Classification Network: Effects of Bandwidth Reduction and Companding / School of Engineering, RMIT University, Melbourne, VIC, Australia, 2020

[4] S. Davis and P. Mermelstein Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. In IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 28 No. 4, 1980.