**Отчет**

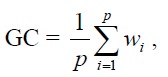
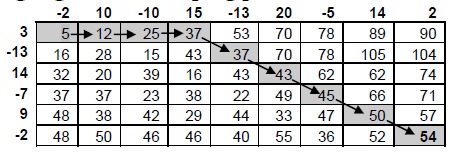
**Теоретическое обоснование**

Для распознавания шаблона в аудиопотоке был реализован алгоритм на основе алгоритма dtw(Dynamic time Wrapping).

**Алгоритм динамической трансформации временно́й шкалы** (*DTW-алгоритм*, от [англ.](https://ru.wikipedia.org/wiki/Английский_язык) *dynamic time warping*) — [алгоритм](https://ru.wikipedia.org/wiki/Алгоритм), позволяющий найти оптимальное соответствие между временными последовательностями. Впервые применен в [распознавании речи](https://ru.wikipedia.org/wiki/Распознавание_речи), где использован для определения того, как два речевых сигнала представляют одну и ту же исходную произнесённую фразу. Впоследствии были найдены применения и в других областях.

[Временные ряды](https://ru.wikipedia.org/wiki/Временной_ряд) — широко распространенный [тип данных](https://ru.wikipedia.org/wiki/Тип_данных), встречающийся, фактически, в любой научной области, и сравнение двух последовательностей является стандартной задачей. Для вычисления отклонения бывает достаточно простого измерения расстояния между компонентами двух последовательностей (евклидово расстояние). Однако часто две последовательности имеют приблизительно одинаковые общие формы, но эти формы не выровнены по оси X. Чтобы определить подобие между такими последовательностями, мы должны «деформировать» ось времени одной (или обеих) последовательности, чтобы достигнуть лучшего выравнивания.

Измерение расстояния между двумя временными рядами нужно для того, чтобы определить их подобие и классификацию. Таким эффективным измерением является евклидова метрика. Для двух временных последовательностей это просто сумма квадратов расстояний от каждой n-ой точки одной последовательности до n-ой точки другой. Однако использование евклидова расстояния имеет существенный недостаток: если два временных ряда одинаковы, но один из них незначительно смещен во времени (вдоль оси времени), то евклидова метрика может посчитать, что ряды отличаются друг от друга. DTW-алгоритм был введён для того, чтобы преодолеть этот недостаток и предоставить наглядное измерение расстояния между рядами, не обращая внимание как на глобальные, так и на локальные сдвиги на временной шкале.

Предположим, что у нас есть две числовые последовательности (a1, a2, …, an) и (b1, b2, …, bm). Как видим, длина двух последовательностей может быть различной. Алгоритм начинается с расчета локальных отклонений между элементами двух последовательностей, использующих различные типы отклонений. Самый распространенный способ для вычисления отклонений является метод, рассчитывающий абсолютное отклонение между значениями двух элементов (Евклидово расстояние). В результате получаем матрицу отклонений, имеющую n строк и m столбцов общих членов:Минимальное расстояние в матрице между последовательностями определяется при помощи алгоритма динамического программирования и следующего критерия оптимизации:где: aij — минимальное расстояние между последовательностями (a1, a2, …, an) и (b1, b2, …, bm). Путь деформации – это минимальное расстояние в матрице между элементами а11 и anm, состоящими из тех aij элементами, которые выражают расстояние до anm.Глобальные деформации состоят из двух последовательностей и определяются по следующей формуле:где: wi – элементы, которые принадлежать пути деформации; p – их количество. Расчеты производились для двух коротких последовательностей и указаны в таблице, в которой выделена последовательность деформации.Существует три условия, налагаемых на DTW алгоритм для обеспечения быстрой конвергенции:  
1. Монотонность – путь никогда не возвращается, то есть: оба индекса, i и j, которые используются в последовательности, никогда не уменьшаются.   
2. Непрерывность – последовательность продвигается постепенно: за один шаг индексы, i и j, увеличиваются не более чем на 1.  
  
3. Предельность – последовательность начинается в левом нижнем углу и заканчивается в правом верхнем.

Для поиска использовались не сами значения сигнала, а Мел-частотные кепстральные коэффициенты (MFCC). MFCC — это своеобразное представление энергии спектра сигнала.

## MFCC

MFCCs обычно вычисляют следующим образом:

1. Вычисляют преобразование Фурье сигнала.
2. Проецируют мощность спектра на  мелшкалу, используя треугольные перекрывающиеся окна.
3. Берут логарифм каждой из частот, получаются мел-коэффициенты.
4. Вычисляют дискретное косинусное преобразование этих коффициентов.
5. MFCC — амплидуты результирующего спектра .

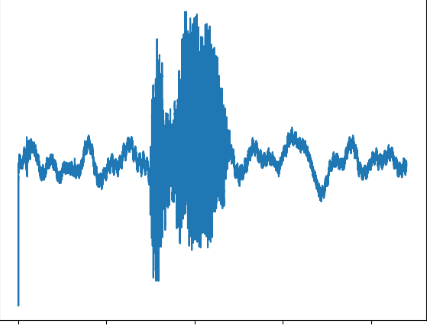
Таким образом алгоритм работает по следующей схеме:

Записывается звук в тишине, звук чиститься от шумов с помощью фильтров для восприятия человеческого голоса (300-3000 гЦ), нормализуется (находится Z-значения его элементов). После этого из звука вырезается часть с записанным словом (для улучшения точности) с помощью нахождения возвышенностей в записываемом в тишине слове. Далее для поиска dtw расстояния находятся mfcc коэффициенты полученного сигнала и он готов для сравнения с записанным после этого звуком.

Записанный вторым звук тоже очищается от шума и нормализуется, после чего находятся его mfcc коэффициенты, но размерность его mfcc будет больше, чем у первого сигнала, так как он дольше. Пусть длина mfcc для первого звука m, для второго n, при этом m <= n, тогда находится dtw дистанция между каждым n-m участком второго сигнала и первым. Далее значимым считаем минимальное расстояние из найденных, которое поможет найти нужное слово во всей второй записи.

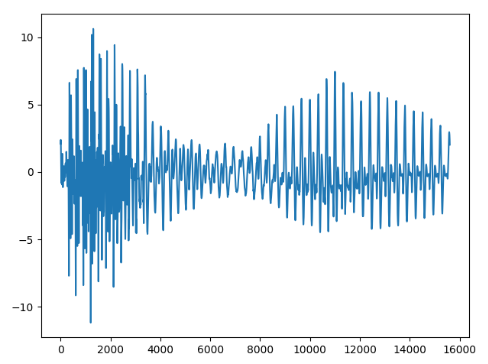
Пример:

В первой записи я в тишине записал слово «один», во второй «один, два». Первая запись графически выглядит так:

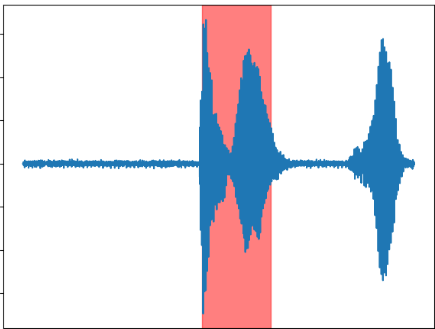


После очистки, нормализации и поиске нужного слова в первой последовательности получим

следующий сигнал:



На рисунке видно, как выглядит слово «один». После необходимых преобразований и поиска дистанции программа показывает во второй записи слово из первой записи, если оно было произнесено:



Видно успешное распознавание необходимого слова.

**Сложность и ограничения**

Алгоритм dtw имеет полиномиальную скорость работы O(m\*n), поэтому запись длится всего две секунды, иначе понадобиться достаточно времени для поиска n-m (здесь n, m размерность mfcc, а не сигнала) расстояний. Так же опытным путем замечено, что алгоритм чувствителен к тембру голоса, поэтому разные голоса на первой и второй записи могут привести к ошибке в поиске. Так же желательно выдерживать небольшую паузу между словами.

**Системные требования:**

При запуске запуске можно выбрать устройство, с которого будет осуществляться запись, поэтому и сама программа работает с помощью микрофона, потому необходимо записывающее устройство, подключенное к компьютеру. Так же программа написана на языку python 3.6.7. Поэтому необходим python 3 и библиотеки librosa, numpy, pyaudio, wave, dtw, matplotlib scipy.

Установить их можно с помощью команды:

pip3 install librosa numpy pyaudio wave dtw matplotlib scipy

Далее необходимо запустить app.py командой

python3 app.py

При желании можно запустить виртуальную среду, чтобы избежать проблемы с зависимостями:

необходимо директорию файлами app.py, recognizer.py открыть в терминале и запустить команды:

pip3 install virtualenv

python3 -m venv venv

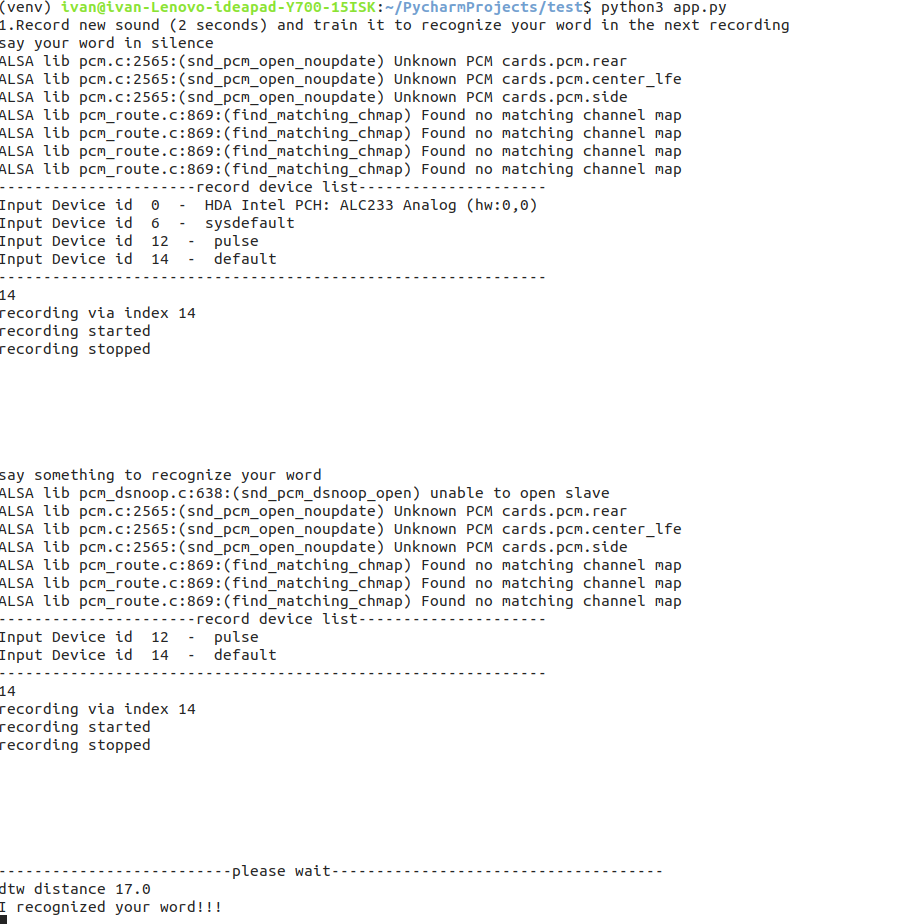
source venv/bin/activate

pip3 install librosa dtw scipy matplotlib numpy wave pyaudio

Для запуска:

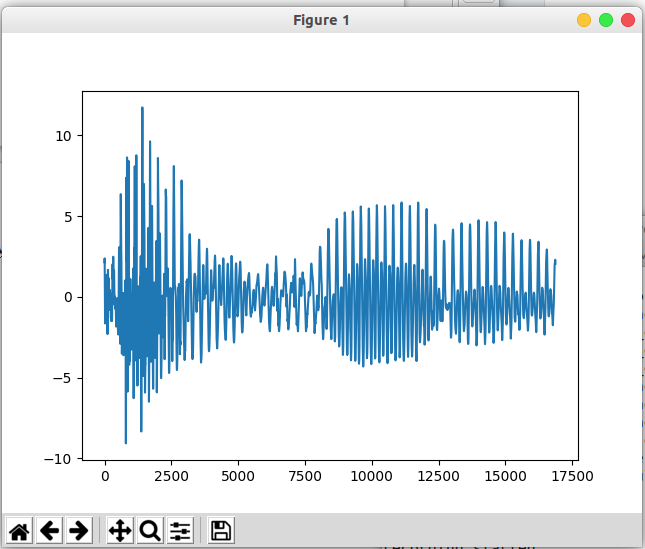
python3 app.py

Пример запуска:



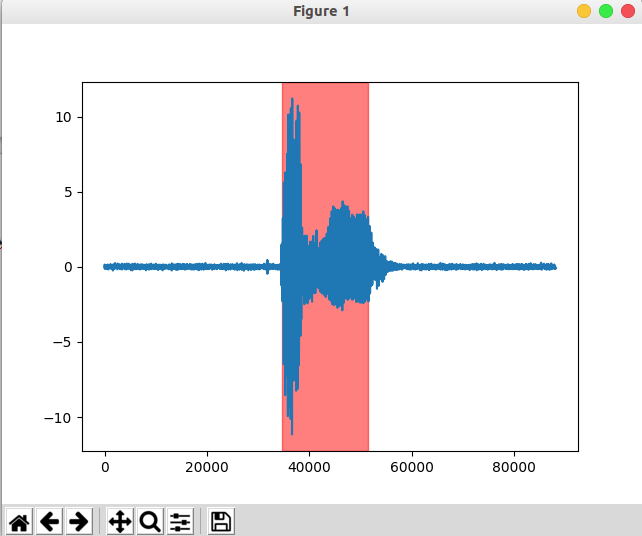
Если приложение выдает строку «I recognized your word!» , то значит, что алгоритм успешно нашел нужное слово. В противном случае приложение выдает строку «you did not say train word», это происходит в случае, если минимальное расстояние меньше определенного порога(эмпирически подобрано 36). Далее приложение показывает график звуковой волны записанного слова.

Изображение входного слова. Далее следует закрыть изображение :



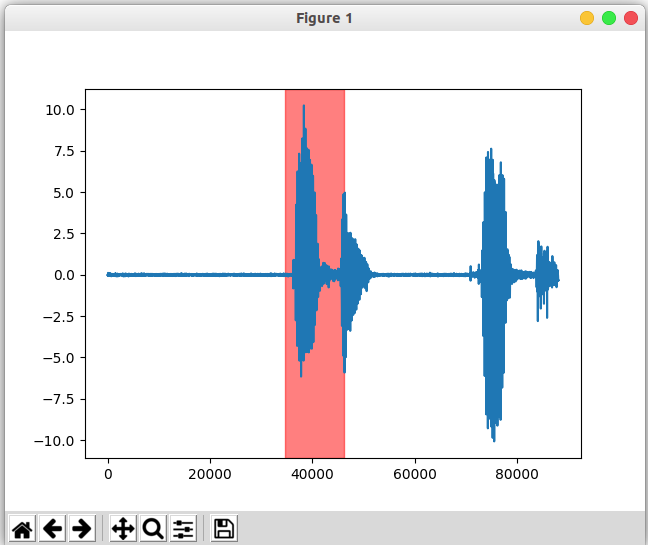
Если слово успешно найдено, то приложение выдаст график аудиопотока с показанным в нем аудиошаблоном.

Результат распознавания слова в заново записанном аудио.

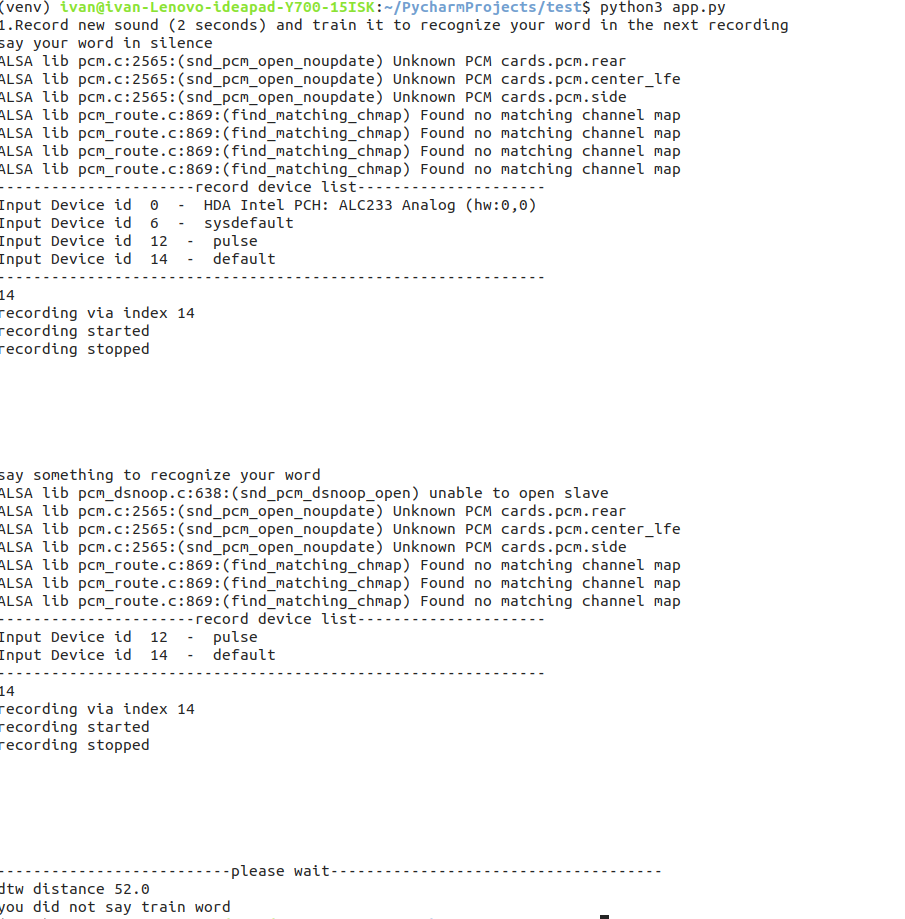


Пример 2

Запись слова «Петя», после запись «Петя, Вася». Выделение шаблонного слова происходит не всегда точно, так как поиск для отображения происходит по наименьшему dtw расстоянию между mfcc коэффициентами, которые не является точным временным отображением.



Пример 3. Пример работы, когда слово найдено не было



Вывод: было создано приложение, распознающее во входящем потоке определённое слово с визуализацией.