Липецкий государственный технический университет

Факультет автоматизации и информатики

Кафедра автоматизированных систем управления

ИНДИВИДУАЛЬНОЕ ДОМАШНЕЕ ЗАДАНИЕ

по дисциплине «Прикладные интеллектуальные системы и экспертные системы»

Распознавание речи и звука

Студент Осипов А.А.

Группа М-ИАП-22

Руководитель Кургасов В.В.

Липецк 2022 г.

Ход работы

Краткая теоретическая справка

Большинство систем распознавания речи построено и развернуто с помощью пакетов и библиотек python. В определенной степени Python доказал, что это важный аспект обозримого будущего. Чтобы включить распознавание речи в Python, потребуется определенный уровень интерактивности и доступности, соответствующий технологиям.

Говоря о составляющих, первая из них – это речь. Она должна быть преобразована из звука в сигнал, который может проходить через микрофон и может быть преобразована в цифровые данные. Это делается с помощью аналого-цифрового преобразователя. После того, как данные оцифрованы, несколько моделей могут легко преобразовать звук в текст.

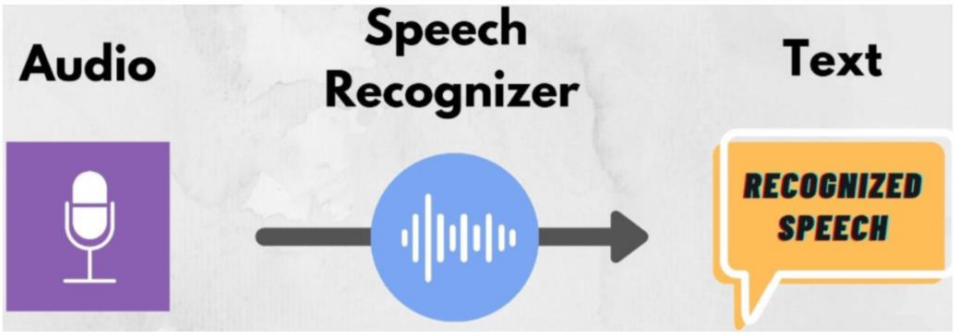

Рисунок 1 – Модель преобразования звука

Программа позволяет распознавать голос и звуковые файлы.

Современные распознаватели речи основаны на менее известной концепции Hidden Markov Model (HMM). Подход основан на предположениях, сформулированных для речевого сигнала, когда он сохраняется в течение короткого периода (скажем, пять миллисекунд), и его можно назвать стационарным процессом, то есть процессом, основанным на статистике, которая не меняется с течением времени.

В типичном HMM разделение речи по умолчанию на сигнал составляет около десяти миллисекунд, разделенных на различные фрагменты. Спектр мощности каждого фрагмента помогает сигналам построить функцию и

сгенерировать частоту, которая позже отображается на вектор действительных чисел, называемый кепстральным коэффициентом.

Группа векторов играет важную роль в декодировании речи в текст с помощью фонем, которые являются основными единицами речи. Вычисление фонем зависит от обучения, так как говорящие различаются, и даже у одного и того же говорящего иногда бывает разное произношение. Поэтому, чтобы справиться с этой проблемой, рассматривается специальный алгоритм, определяющий наиболее относительные слова, образующие последовательность фонем.

Вы когда-нибудь задумывались о том, как Google Assistant или Amazon Alexa распознает все, что вы говорите? Помимо огромного успеха на рынке систем распознавания, большая часть сотовых устройств имеет функцию распознавания речи через некоторые встроенные приложения или сторонние приложения.

ADRIVER

Не обязательно, но большинство таких систем распознавания речи построено и развернуто с помощью пакетов и библиотек python. В определенной степени Python доказал, что это важный аспект обозримого будущего. Причина довольно очевидна. Чтобы включить распознавание речи в Python, вам потребуется определенный уровень интерактивности и доступности, соответствующий технологиям.

Концепция доступности заслуживает рассмотрения, потому что распознавание позволяет пожилым людям, людям с ограниченными физическими возможностями или слабовидящим людям взаимодействовать с машинами и быстро решать свои проблемы с помощью современных услуг и продуктов, не выбирая случайные приложения с графическим интерфейсом пользователя.

В этой статье мы научимся создавать систему распознавания речи и голоса на Python. Чтобы облегчить процесс понимания того, как она устроена, эта статья призвана научить вас, как построить систему с меньшими усилиями

и с большим энтузиазмом. Но прежде чем перейти к проекту, давайте поговорим о некоторых не менее важных аспектах, которые нужно знать разработчику.

Как работает распознавания речи?

Прежде чем переходить к деталям и сложностям проекта, вы должны составить и понять краткий обзор работы распознавания речи. Хотя для начала работы нет никаких предварительных условий, все же хорошо владеть основами языка программирования Python.

РЕКЛАМА

Говоря о составляющих, первая из них – это речь. Она должна быть преобразована из звука в сигнал, который может проходить через микрофон и может быть преобразована в цифровые данные. Это делается с помощью аналого-цифрового преобразователя. После того, как данные оцифрованы, несколько моделей могут легко преобразовать звук в текст.

Современные распознаватели речи основаны на менее известной концепции Hidden Markov Model (HMM). Подход основан на предположениях, сформулированных для речевого сигнала, когда он сохраняется в течение короткого периода (скажем, пять миллисекунд), и его можно назвать стационарным процессом, то есть процессом, основанным на статистике, которая не меняется с течением времени.

В типичном HMM разделение речи по умолчанию на сигнал составляет около десяти миллисекунд, разделенных на различные фрагменты. Спектр мощности каждого фрагмента помогает сигналам построить функцию и сгенерировать частоту, которая позже отображается на вектор действительных чисел, называемый кепстральным коэффициентом.

Размеры отображаемого вектора довольно малы, всего 10, по сравнению с некоторыми точными системами, которые могут иметь размеры, достигающие 32 или более. Сгенерированный окончательный результат HMM принимает форму векторных последовательностей.

Группа векторов играет важную роль в декодировании речи в текст с помощью фонем, которые являются основными единицами речи. Вычисление фонем зависит от обучения, так как говорящие различаются, и даже у одного и того же говорящего иногда бывает разное произношение. Поэтому, чтобы справиться с этой проблемой, рассматривается специальный алгоритм, определяющий наиболее относительные слова, образующие последовательность фонем.

Кроме того, детекторы голосовой активности (VAD) также используются для уменьшения некоторой части аудиосигнала, которая может содержать речь. В основном он используется для распознавания неважных частей речи и предотвращения их принятия во внимание.

Пакеты распознавания речи

В цепочке PyPI существует несколько пакетов для распознавания речи. Вот некоторые из них:

- Assembly
- Apia
- SpeechRecognition
- Wit
- Watson-developer-cloud

Вышеуказанные пакеты, такие как apiai и wit, предлагают функцию обработки естественного языка. Эта встроенная функция помогает определить намерения говорящего и выходит за рамки обычного распознавания речи. Другие пакеты в основном ориентированы на преобразование речи в текст.

Для распознавания речи требуется некоторый ввод в виде звука, и пакет SpeechRecognition легко извлекает этот тип ввода. Для доступа к микрофонам и последующей обработки звука с нуля не требуются сложные скрипты. Еще одним преимуществом этого пакета является то, что может выполнить инструкции за несколько минут.

Библиотека SpeechRecognition ведет себя как обложка или оболочка для различных API, созданных исключительно для речи. Он невероятно гибкий и

маневренный. Одним из таких API является API Google Web Speech, который поддерживает жестко заданное распознавание речи по умолчанию.

Библиотека SpeechRecognition очень проста в использовании, а пакет легко импортировать как проект python. Также важно отметить, что этот пакет может не включать в себя все API, доступные сегодня. Таким образом, вам нужно точно определить, какой пакет вам нужен для создания распознавателя речи.

Основная цель Recognizer – распознавать речь вместе с вариантами чтения различных речей, а затем управлять функциями и проверять речь, исходящую от источника звука.

Перед тем как приступить к работе с пакетом SpeechRecognition в Python, необходимо загрузить аудиофайл. SpeechRecognition упрощает работу с аудиофайлами, сохраняя их в том же каталоге интерпретатора Python. Это делается с помощью класса AudioFile. Этот класс необходимо инициализировать и указать путь к аудиофайлу, чтобы диспетчер контекста предоставлял удобный интерфейс для чтения файлов и их содержимого.

Типы форматов файлов, которые поддерживает SpeechRecognition, указаны ниже:

WAV: формат должен быть в PCM / LPCM

AIFF

AIFF-C

FLAC: формат должен быть собственным FLAC

Функция record() используется для захвата данных из файла с помощью интерпретатора python в файле.

При использовании внутри блока, метод record() всегда намеревается продвигаться вперед в файловом потоке. Обычно это означает, что запись повторяется в течение четырех секунд и возвращает первый четырехсекундный звук.

Пример выполнения задания

Преобразование звука в текст

```
with file as source:
 r.adjust_for_ambient_noise(source)
 audio = r.record(source)
 result = r.recognize_google(audio,language='ru')
print(result)
```

```
result2:
{   'alternative': [   {   'confidence': 0.95275849,
                           'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидение '
                                         'смотрите И слушайте Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте в Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москва'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москвы'}],
    'final': True}
говорит и показывает Москва работают все центральные каналы телевидения смотрите И слушайте Москву
```

Рисунок 2 – Пример выполнения задания

Создадим 7 тестовых заданий

```
In [10]: import speech_recognition as sr
         r = sr.Recognizer()
```

```
In [13]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=10)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)
```

```
Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {   'confidence': 0.95275831,
                           'transcript': 'подводная лодка утонула'}],
    'final': True}
подводная лодка утонула
```

Рисунок 3 – Результаты работы программы

```
In [22]: import speech_recognition as sr
         r = sr.Recognizer()
```

```
In [23]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=10)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)
```

```
Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {'confidence': 0.81742543, 'transcript': 'Салават'},
                       {'transcript': 'самовар'},
                       {'transcript': 'помогает'},
                       {'transcript': 'омагад'},
                       {'transcript': 'полноват'}],
    'final': True}
Салават
```

Рисунок 4 – Результаты работы программы

```
In [5]: with file as source:
         r.adjust_for_ambient_noise(source)
         audio = r.record(source)
         result = r.recognize_google(audio,language='ru')
        print(result)

result2:
{   'alternative': [   {   'confidence': 0.95275849,
                           'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидение '
                                         'смотрите И слушайте Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте в Москву'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москва'},
                       {   'transcript': 'говорит и показывает Москва работают '
                                         'все центральные каналы телевидения '
                                         'смотрите И слушайте Москвы'}],
    'final': True}
говорит и показывает Москва работают все центральные каналы телевидения смотрите И слушайте Москву
```

Рисунок 5 – Результаты работы программы

```
In [29]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=10)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)

Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {   'confidence': 0.95275831,
                           'transcript': 'Шла Саша по шоссе и сосала сушку'}],
    'final': True}
Шла Саша по шоссе и сосала сушку
```

Рисунок 6 – Результаты работы программы

```
In [32]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=10)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)

Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {   'confidence': 0.95275831,
                           'transcript': 'Простите пожалуйста'},
                       {'transcript': 'Прости пожалуйста'},
                       {'transcript': 'Простите пожалуйста меня'},
                       {'transcript': 'Простите пожалуйста -'},
                       {'transcript': 'Простите пожалуйста Лена'}],
    'final': True}
Простите пожалуйста
```

Рисунок 7 – Результаты работы программы

```
In [34]: import speech_recognition as sr
         r = sr.Recognizer()

In [35]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=5)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)

Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {'confidence': 0.67460966, 'transcript': 'Здорово'},
                       {'transcript': 'здорова'},
                       {'transcript': 'Здарова'},
                       {'transcript': 'Здарово'}],
    'final': True}
Здорово
```

Рисунок 8 – Результаты работы программы

```
In [34]: import speech_recognition as sr
         r = sr.Recognizer()

In [36]: with sr.Microphone() as source:
          r.adjust_for_ambient_noise(source)
          data = r.record(source, duration=5)
          print('Sesinizi Tanımlıyor…')
          text = r.recognize_google(data,language='ru')
          print(text)

Sesinizi Tanımlıyor…
result2:
{   'alternative': [   {   'confidence': 0.94999379,
                           'transcript': 'наступающим Новым годом'},
                       {'transcript': 'С наступающим Новым годом'}],
    'final': True}
наступающим Новым годом
```

Рисунок 9 – Результаты работы программы

Код программы

```
pip install SpeechRecognition
import speech_recognition as sr
r = sr.Recognizer()
file = sr.AudioFile('Sound.wav')


with file as source:
 r.adjust_for_ambient_noise(source)
 audio = r.record(source)
 result = r.recognize_google(audio,language='ru')
print(result)


pip install pyaudio
import speech_recognition as sr
r = sr.Recognizer()


with sr.Microphone() as source:
 r.adjust_for_ambient_noise(source)
 data = r.record(source, duration=10)
 print('Sesinizi Tanımlıyor…')
 text = r.recognize_google(data,language='ru')
 print(text)


import speech_recognition as sr
import pyttsx3
import pywhatkit
import datetime
import wikipedia
import pyjokes

listener = sr.Recognizer()
engine = pyttsx3.init()
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)

def talk(text):
    engine.say(text)
    engine.runAndWait(

def take_command():
    try:
        with sr.Microphone() as source:
            print('listening...')
            voice = listener.listen(source)
            command = listener.recognize_google(voice)
            print(command)
    except:
        pass
return command

def run_alexa():
    command = take_command()
    print(command)
    if 'play' in command:
```

```
        song = command.replace('play', '')
        talk('playing ' + song)
        pywhatkit.playonyt(song)
    elif 'time' in command:
        time = datetime.datetime.now().strftime('%I:%M %p')
        talk('Current time is ' + time)
    elif 'who the heck is' in command:
        person = command.replace('who the heck is', '')
        info = wikipedia.summary(person, 1)
        print(info)
        talk(info)
    elif 'date' in command:
        talk('sorry, I have a headache')
    elif 'are you single' in command:
        talk('I am in a relationship with wifi')
    elif 'joke' in command:
        talk(pyjokes.get_joke())
    else:
        talk('please say the command again')

while True:
    run_alexa()
```

Вывод

В ходе выполнения индивидуального домашнего задания мы получили базовые навыки работы с языком python и набором функций для анализа и обработки данных.