Софийски университет „Св. Кл. Охридски”

Факултет по математика и информатика

Курсов Проект

на тема: „Разпознаване на Говор и Превод”

Студент: **Стела Тодорова Маринова** Ф.Н. **81585**

Студент: **Виктор Велинов Русев** Ф.Н. **81644**

Курс: „4“, Учебна година: 2021/22

Преподавател: **проф. Иван Койчев**

==========

Декларация за липса плагиатство:

- Плагиатство е да използваш, идеи, мнение или работа на друг, като претендираш, че са твои. Това е форма на преписване.
- Тази курсова работа е моя, като всички изречения, илюстрации и програми от други хора са изрично цитирани.
- Тази курсова работа или нейна версия не са представени в друг университет или друга учебна институция.
- Разбирам, че ако се установи плагиатство в работата ми ще получа оценка “Слаб”.

11.2.22 г. Подпис на студента:

Съдържание

1 Увод

Автоматично генериране на субтитри на различни езици от звуков или видео файл.

2 Основни задачи

Задачата се разбива на две подзадачи – разпознаване на говор и превод на текст.

2.1 Разпознаване на говор

Идеята е от подаден звуков или видео файл да се генерира текстов файл със субтитри, който в последствие да бъде преведен от програмата.

От входния файл се извлича звука и се разбива на части. За всяка част се генерира спектограма. Те биват обработени от конволюционна невронна мрежа, която е предварително обучена да разпознава говор чрез „алчен“ декодиращ агент.

Крайният резултат от тази задача е да се получи един .srt файл с транскрипция на звука от входния видео файл.

Файлът се използва в следващата стъпка като вход.

2.2 Превод на текст

Тази част от програмата разчита на подаден текстов файл със генерирани субтитри и информация за поредност и време през което субтитрите са „видими“ на екрана. Форматът на редовете трябва да е както следва:

- Поредност (индекс) на блок текст
- Време на „видимост“
- Текст
- Празен ред

Файлът се разчита от програмата и всеки отделен ред „текст“ бива „преведен“ от модел, който предварително е обучен на входен и изходен език, заедно с тяхната граматика. Програмата може да приема повече от един изходен език.

Моделът е базиран на embedding двупосочна рекурентна невронна мрежа (Embedding Bidirectional RNN).

За целта, всяко парче „текст“ бива предварително подготвено, за бъде съвместимо с модела. То преминава през фаза на токенизиране и допълване на „празни“ думи до стигане на броя думи в най дългото изречение от модела.

Крайният резултат от тази задача е да се получи един .srt с име, започващо с името на подадения на входа файл с префикс езика, на който е преведен.

Този файл може да бъде зареден в оригиналния звуков или видео файл.

3 Проектиране

За система/приложение: На кратко: Анализ на изискванията, Обща архитектура – напр. слоеве, модули, блокове, компоненти...; Модел на данните; Схема за представяне на знанията. Диаграми; Потребителски интерфейс (ако има); Ресурсни;…

За Ако има **изследователски проекти:** Изследователски хипотези; Данни; Планиране на експерименти.

4 Реализация / технологии, платформи и библиотеки

4.1 Разпознаване на говор

Извличане на аудио от видео с помощта на библиотеката *moviepy*. Целта е създаване на звуков файл с подобно име на това на видео файла в .wav формат.

Разбиване на звуковия файл на части с помощта на библиотеката *pydub.* Локализираме паузите в звуковия файл. За пауза дефинираме липса на говор/шум за поне 500 милисекунди:

1. Извличаме начално и крайно време в милисекунди за всяко отделно парче звук между паузите
2. Записваме всяко отделно парче звук като нов *.wav* файл и приготвяме шаблонен *.srt* файл за него с номер и начално и крайно време в *hh:mm:ss.mss* формат

Разпознаване на говор с помощта на библиотеките *pytorch* и *torchaudio.* Тази стъпка включва подготовка на тренировъчно множество:

1. Използвана база данни със свободен достъп *Librispeech* и по-конкретно *train-clean-100*
2. Разделяне на подмножества с размер 32 записа. Всеки запис се състои от звуков файл в *.flac* формат и текстов файл, съдържащ транскрипция на звука. За всеки запис създаваме спектограма и етикет.
   2.1. Спектограмата съдържа маскирани (изтрити) данни по време и по честота
   2.2. Етикетът се сформира като всеки символ от текста се преобразува до цяло число

Невронната мрежа се състои от:

1. Спектограма
2. Конволюционна мрежа
3. Остатъчна Конволюционна мрежа
   3.1. Подобрява скоростта на модела и класификацията му чрез „изглаждане“ и намаляване на загубите. По този начин моделът по-бързо намира минимуми и съответно - решение
   3.2. Двупосочна остатъчна конволюционна мрежа с *Gated Recurrent Unit (GRU).* Изисква по-малко пресмятания в сравнение с други алтернативи
   3.3. Всичко се базира на *pytorch* невронна мрежа
   3.4. Оптимизатор – *AdamW*
   3.5. Планер – *One Cycle Learning Rate*

Обучение на модела - за разлика от много подобни модели, нашата цел е да можем да обработим неразпознати звуци (шум). Тъй като валидните символи са отбелязани в числов код от 0 до 27, то логично невалиден символ / звук ще бележим с код 28. Методът *CTCLost* от *pytorch* ни дава механизъм за тази обработка. Веднъж обучен, моделът може да бъде запазен и използван многократно.

Предсказването на данни се случва с помощта на „алчен“ декодиращ агент. Той приема матрица на вероятностите за всеки символ като за всеки кадър от спектограмата избира етикета с най-голяма вероятност. При неуспешно разпознаване на символа (празен / невалиден символ), той бива премахнат от финалния транскрипт. Агентът обработва всички парчета от аудио файла и записва транскрипцията в съответния шаблон за субтитри.

След приключване на изпълнението се очаква да разполагаме с един *.srt* файл с транскрипцията на звука от входния видео файл

Технологии

1. *pytorch* и *torchaudio*
    1.1. Мощни библиотеки за машинно самообучение на езика *Python* и под-библиотека за работа с анализ на звук, свободна за ползване
    1.2. Мултинишково процесорно изпълнение с паралелизация
    1.3. Възможност за съхранение на вече обучен модел и повторното му използване
    1.4. Множество вградени алгоритми
    1.5. По-добро представяне откъм време спрямо алтернативата *Keras*
    1.6. Алгоритъм *AdamW,* който е един от предпочитаните алгоритми за класификация на звук. Алтернатива: стохастично градиентно спускане, но макар и да се представя по-добре с класифицирането, *AdamW* печели по-добро време и компютърна мощност
2. *CometML*
    2.1. Система за мониторинг на представянето на даден модел
    2.2. Има интеграция с *pytorch*
3. *Pydub*
    3.1. Библиотека за работа със звук
    3.2. По-лека за използване от алтернативната *librosa*

4.2 Превод на текст

След предварително обучена рекурентна невронна мрежа с помощта на *Keras*, целта на тази задача е да преведе субтитрите от оригиналния език на избран от потребителя език като моделът се съобразява с граматиката на изходния език.

Програмата може да превежда на няколко езика последователно. Обучаването на модела разчита на набор от двойки от речници – съответно на входния език и на изходните, определени от клиента. Реализацията следва няколко стъпки:

1. Зареждане на двойка речници – оригиналния език и поредния изходен език

2. подготвяне на данните за обучението на модела с помощта на класа *Tokenizer* от *Keras*
    2.1. токенизация - създава се речник, в който на всяка дума се задава уникален индекс - цяло число
    2.2. уеднаквяване размера на разширяване на данните (*padding*)
    2.3. като бонус към задача на стандартния изход (в конзолата) се представя и сложността на конкретен речник - от колко думи се състои, колко са уникалните думи, кои са десетте най-често срещани думи, какъв е размерът на най-дългото изречение, както и по две примерни изречения от всеки език

```
--- Complexity of data
english 1823250 words.
227 unique english words.
10 Most common words in the english dataset:
"is" "," "." "in" "it" "during" "the" "but" "and" "sometimes"

french 1961295 words.
354 unique french words.
10 Most common words in the french dataset:
"est" "." "," "en" "il" "les" "mais" "et" "la" "parfois"

Max english sentence length: 15
Max french sentence length: 21
english vocabulary size: 199
french vocabulary size: 344
```

3. В програмата са реализирани общо 5 модела – Еднопосочен Линеен *RNN*, Двупосочен *RNN*, *Embedding RNN*, *Encode-Decode RNN* и *Embedding-Bidirectional RNN*
    3.1. В примерите е използван *Embedding-Bidirectional RNN*

Следващата стъпка от програмата е разчитане на входния файл и зареждането му в модела. Целта е с помощта на вече обучения модел да се преведат изреченията от оригиналния език. Това се случва в следните стъпки:

1. Прочитане на оригиналния файл
2. Всяко изречение бива обработено преди да се подаде към модела за превод. Това включва токенизация и добавяне на „празни“ думи до достигане броя думи на най-дългото изречение в модела
3. Подаване на масив от всички обработени изречения към модела

Като последна стъпка е генериране на финален файл, който съдържа оригиналните индекси, времетраения на всяко парче текст и преведен текст на желания език. Всичко това се случва се случва със стандартни подходи на езика *Python.*

4.3 Реализация на модули

Проектът е разделен на два основни модула – съответно за двете отделни задачи.

Всеки модул е самостоятелно работещ.

5 Заключение

Обобщение на направеното/резултатите.

Идеи за по-нататъшно развитие, усъвършенстване или други експерименти.

Генерирани субтитри от първата задача

speech.srt

```
1
00:00:00.000 --> 00:00:03.245
Hi Jacko it's Pete I am just wandering where you are

2
00:00:03.747 --> 00:00:04.485
I mean Um

3
00:00:12.053 --> 00:00:13.809
so we got it for the survey where
```

Преведени субтитри на френски

speech_french.srt

```
1
00:00:00.000 --> 00:00:03.245
chine est chaud en printemps mais il est chaud en en

2
00:00:03.747 --> 00:00:04.485
il est est en en dernier

3
00:00:12.053 --> 00:00:13.809
il est quil en en decembre
```

6 Разпределение на задачи

Разпознаване на Говор и архитектура – **Стела Маринова**

Превод на Текст и архитектура – **Виктор Русев**

Презентация – **Стела Маринова**

Документация – **Виктор Русев**

7 Използвана литература

Разпознаване на Говор

https://www.assemblyai.com/blog/end-to-end-speech-recognition-pytorch/
https://www.redhat.com/architect/speech-recognition-tips
http://mmsip.bas.bg/publ/CNN_Dimov.pdf
https://medium.com/@magodiasanket/implementation-of-neural-machine-translation-using-python-82f8f3b3e4f1

Превод на Текст

https://stackabuse.com/python-for-nlp-neural-machine-translation-with-seq2seq-in-keras/
https://www.analyticsvidhya.com/blog/2019/01/neural-machine-translation-keras/
https://towardsdatascience.com/neural-machine-translation-with-python-c2f0a34f7dd