Digital Speech Processing Homework #3

ZhuYin Decoding

黃健祐

November 13, 2019

Due on December 6, 2019

Outline

1. Introduction
2. SRILM Toolkit
3. Homework
4. Requirements
5. Grading
6. Contact TAs

Introduction

ZhuYin Decoding

ZhuYin is used widely in our daily life.

- 要不要ㄔ飯呀 → 要不要吃飯呀
- 我已經吃飽ㄌ → 我已經吃飽了
- 想ㄏ珍奶 → 想喝珍奶
- 我在減肥，ㄅ喝含糖飲料 → 我在減肥，不喝含糖飲料
- ㄈㄓ買ㄐ排配無糖綠茶 → 肥宅買雞排配無糖綠茶

In speech recognition, imperfect acoustic models with phoneme loss yield similar result, which we have to correct by post-processing.

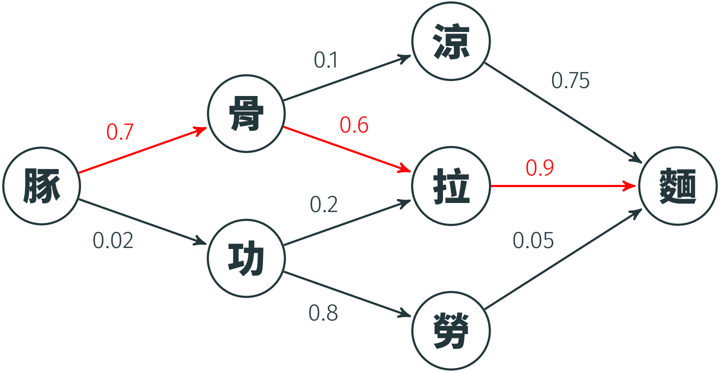
Language Model

Sentences can be reconstructed with the help of **language model**.

$$
\begin{aligned}
W^* &= \underset{W}{\operatorname{argmax}} \, \mathrm{P}(W \mid Z) && (1)\\
&= \underset{W}{\operatorname{argmax}} \, \frac{\mathrm{P}(W)\,\mathrm{P}(Z \mid W)}{\mathrm{P}(Z)} && (2)\\
&= \underset{W}{\operatorname{argmax}} \, \mathrm{P}(W)\,\mathrm{P}(Z \mid W) && (3)\\
&= \underset{W}{\operatorname{argmax}} \, [\mathrm{P}(w_1) \prod_{i=2}^{n} \mathrm{P}(w_i \mid w_{i-1})][\prod_{i=1}^{n} \mathrm{P}(z_i \mid w_i)] && (4)\\
&= \underset{W,\ \mathrm{P}(Z|W)\neq 0}{\operatorname{argmax}} \, [\mathrm{P}(w_1) \prod_{i=2}^{n} \mathrm{P}(w_i \mid w_{i-1})] && (5)
\end{aligned}
$$

Viterbi Algorithm

Viterbi algorithm gives the path with the greatest probability.
For example, 豚ㄍㄌ麵



SRILM Toolkit

SRILM Toolkit

- SRILM Toolkit stands for SRI Language Modeling Toolkit.
- It is a toolkit for building and applying various statistical language models.
- It provides useful C++ libraries, which we are going to exploit in this homework.
- You can either compile from source code on your own or download pre-built docker image.

Setup SRILM

While you can either compile from source code or download pre-built docker image, we recommend you choose the latter one since it is similar to TA's environment.

For pre-built docker image, simply use the following command:

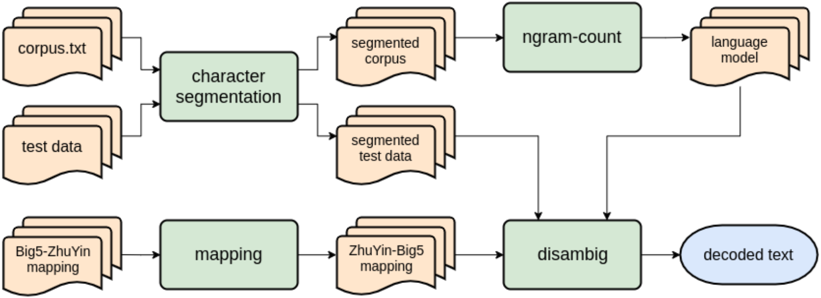
```
docker run -it ntudsp2019fall/srilm
```

As for compiling on your own, please refer to FAQ[1].

[1] http://speech.ee.ntu.edu.tw/DSP2019Autumn/hw3/FAQ.html

Homework

Workflow


corpus.txt
test data
character segmentation
segmented corpus
segmented test data
ngram-count
language model
Big5-ZhuYin mapping
mapping
ZhuYin-Big5 mapping
disambig
decoded text

Provided Files

Big5-ZhuYin.map

- a mapping from Chinese character to ZhuYin

corpus.txt

- corpus data

Makefile

- Makefile template

separator_big5.pl

- script used to separate Chinese words

setup.sh

- script used to activate SRILM environment

test_data

- directory for test data

```
dsp_hw3
├── Big5-ZhuYin.map
├── corpus.txt
├── Makefile
├── separator_big5.pl
├── setup.sh
└── test_data
    └── ...
```

Setup

To activate SRILM environment, simply use the provided script:

```
source setup.sh
```

You have to modify `SRILM_REP_PATH` and `MACHINE_TYPE` on your own if not using provided docker image.

Character Segmentation

Separate Chinese words into characters:

```
perl separator_big5.pl $1 > $2
```

$1 input text file

$2 output segmented file

- 三玖 天下 第一 → 三 玖 天 下 第 一
- 愛 吃 拉麵 的 小泉 同學 → 愛 吃 拉 麵 的 小 泉 同 學
- 數位 語音 處理 概論 → 數 位 語 音 處 理 概 論

Language Model

Build the language model:

```
ngram-count -text $1 -write $2 -order $3
```

$1	input text file
$2	output count file
$3	order of n-gram

```
ngram-count -read $1 -lm $2 -order $3 -unk
```

$1	input count file
$2	output language model file
$3	order of n-gram
unk	view OOVs as <unk> instead of removing them

Language Model Format

- count file

```
夏  11210
俸  267
鴟  7
祇  1
微  11421
```

- language model file

```
\data\
ngram 1 = 6868
ngram 2 = 1696830
\1-grams:
-1.178429 </s>
-99 <s> -2.738217
-1.993207 一 -1.614897
```

ZhuYin-Big5 Map

Create a ZhuYin-Big5 mapping from Big5-ZhuYin mapping:

Big5-ZhuYin		ZhuYin-Big5	
一	ㄧˊ / ㄧˋ / ㄧ _	ㄅ	八 匕 卜 …
乙	ㄧˇ	八	八
丁	ㄉㄧㄥ _	匕	匕
	…		…
長	ㄔㄤˊ / ㄓㄤˇ	ㄆ	仆 匹 片 丕 …
行	ㄒㄧㄥˊ / ㄏㄤˊ	仆	仆
	…		…

- Be aware of **polyphones** (破音字).
- There could be arbitrary spaces between all characters.
- The order can be arbitrarily permuted.

Decoding

Decode the data by:

1. disambig provided by SRILM

   ```
   disambig -text $1 -map $2 -lm $3 -order $4 > $5
   ```

$1	segmented file to be decoded
$2	ZhuYin - Big5 mapping
$3	language model
$4	order of language model
$5	output file

2. your own disambig (MyDisambig)
   - Please implement it by Viterbi algorithm.
   - Bigram is required while trigram is for bonus.
   - Details will be described in `Requirements`.

Requirements

Makefile Format

Your programs should be compiled by Makefile with:

```
make SRIPATH=$1 MACHINE_TYPE=$2
```

$1 SRILM path

$2 machine type

In provided docker image, `SRIPATH` is `/root/srilm-1.5.10` and `MACHINE_TYPE` is `i686-m64`.

At least one executable named `mydisambig` should be compiled.

Mapping Program Format

You are allowed to use **C**, **C++**, or **Python** in this part. TA will run your program by the following command:

```
make map FROM=$1 TO=$2
```

$1 Big5-ZhuYin mapping file (input)

$2 ZhuYin-Big5 mapping file (output)

If you are using **C** or **C++**, your mapping program should be compiled in this or previous step.

MyDisambig Format

You are required to implement your own disambig using C++. It will be executed by the following command:

```
./mydisambig $1 $2 $3 $4
```

$1	segemented file to be decoded
$2	ZhuYin-Big5 mapping
$3	language model
$4	output file

The output format of your program should be **exactly the same as** that of SRILM disambig.

SRILM Output Format

Here is an example of SRILM disambig output:

```
<s> 聲 優 鈴 木 實 里 </s>
```

- There are always a `<s>` at the beginning and a `</s>` at the end.
- There is exactly one space between each token.

Hints

- A Chinese character in Big5 is always encoded with 2 bytes, namely, `char[2]` in `C++`.
- Be careful not to change the encoding of data, otherwise your program may fail to parse Big5 encoded files when grading.
- Here are some SRILM libraries you may find it useful:
    - `$SRIPATH`/include/File.h
    - `$SRIPATH`/include/Ngram.h
    - `$SRIPATH`/include/Vocab.h

Report Format

Write a report (at most **two pages**) in **PDF** format, name it `report.pdf` and submit with your source code.

Your report should at least include the following contents:

- your name and student ID
- what you observed (e.g., disambig vs. MyDisambig)
- what you have done (e.g., trigram decoding)

If implementing trigram decoding, you should describe it in a detailed manner. The bonus point will be granted according to both program performance and description in report.

Submission Format

Your submission should be a **ZIP** file and be uploaded to CEIBA. Its file structure should be the same as the following:

```
<zip_file>
└── hw3_<student_id>
    ├── Makefile
    ├── report.pdf
    ├── inc
    │   └── [*.h, *.hpp]
    └── src
        └── [*.c, *.cc, *.cpp, *.py]
```

- All of your source code files must be placed under `inc` and `src`.
- **DO NOT** create any subdirectory in `inc` and `src`.

Grading

Grading Method & Environment

Your mapping and disambig program will be tested respectively.

Mapping is allowed to run for 30 seconds while MyDisambig is required to finish in 1 minute for each input data. Programs will be interrupted when time limit exceeds.

The environment TA will use is the same as the provided docker image except for **different** SRILM path.

Grading Policy

$$\mathrm{SCORE} = (F + M) \times (\mathrm{Mapping} + \mathrm{MyDisambig} + \mathrm{Report} + \mathrm{Trigram})$$

Item	Score	Description
File Format	0%	correct file format → $F = 0.75$, otherwise $F = 0$.
Make	0%	makable → $M = 0.25$, otherwise $M = 0$.
Mapping	15%	Correctly generating mapping in time for full credit, otherwise you will get 0.
MyDisambig	70%	35% for successful execution in time, and 35% for accuracy.
Report	15%	You will get 0 if it is more than 2 pages.
Trigram	10%	Bonus for trigram MyDisambig.

Late Submission

Due on December 6, 2019

You are still allowed to submit after the due date. The penalty for late submission is an exponential decay with decay rate 1.5%[2] of the maximum grade applicable for the assignment, for each hour that the assignment is late.

An assignment submitted more than 3 days after the deadline will have a grade of zero recorded for that assignment.

$$\mathrm{SCORE}_{final}(\mathrm{hr}) = \begin{cases} \mathrm{SCORE}_{original} \times 0.985^{\mathrm{hr}} & , \ \mathrm{hr} \leqslant 72 \\ 0 & , \ \mathrm{hr} > 72 \end{cases}$$

[2] less than 70% after 24 hrs, 48% for 48 hrs and 33% for 72 hrs

DO NOT CHEAT

Any form of cheating, lying, or plagiarism will not be tolerated.

Contact TAs

Should you have any question or need help,

- please read the FAQ[3] first.
- send email to *ntudsp_2019fall_ta@googlegroups.com* with "**`[HW3]`**" as the subject line prefix.
- or come to EE2 R531 with prior notification.

黃健祐	Mon.	10:00 - 12:00
	Fri.	14:00 - 17:00

Office hours

[3] http://speech.ee.ntu.edu.tw/DSP2019Autumn/hw3/FAQ.html