CISC7021 - Applied Natural Language Processing - Assignment 2

Zhang Huakang
M-C3-5095-0

1 Environment Setup

In this project, four software packages are used: **Boost**, which is used for providing C++ libraries that are useful for tasks such as linear algebra, multithreading; **cmph**, which is a C library for minimal perfect hashing; **mosesdecoder**, which is a statistical machine translation system, and **mgiza**, which is a multi-threaded implementation of the word alignment tool GIZA++.

The installation is done on a docker container with Ubuntu 14.04. This *Dockerfile* is published on Github.

Since the unfamiliarity of the software packages, I spent a lot of time on the installation.

Because the tokenization of Chinese is different from English, I use *ansjTokenizer* to tokenize Chinese, which is implemented by Java. *openjdk:8-jdk* image is used to run the Java program.

2 Tokenize

Listing 1: Chinese Tokenization

```
openjdk:8-jdk \
java -jar /mnt/ansjTokenizer.jar \
/mnt/train.tags.zh-en.zh \
/mnt/train.token.zh
```

Listing 2: English Tokenization

```
$MOSES_TOKEN/tokenizer.perl\
-l en \
-threads 4 \
< /mnt/train.tags.zh-en.en \
> /mnt/train.token.en
```

Listing 3: Reduce Parallel Corpus

```
$MOSES_TRAINING$/clean-corpus-n.perl\
/mnt/train.token \
en \
zh \
/mnt/train.token.clean.50 \
1 \
50 \
-lowercase 1
```

By the code in *Listing 3*, we limit the length of the parallel corpus text to no more than fifty words, and convert the English to all lowercase.

Such pre-processing is also done for the test and dev data.

3 3-gram Language Model

LMPLZ is used to train the language model, which is estimates language models with Modified Kneser-Ney smoothing and no pruning. The command is shown in *Listing 4*.

Listing 4: Build Language Model

```
$MOSE_BIN/lmplz \
-o 3 \
-S 50% \
-T /mnt/tmp \
--text /mnt/train.token.clean.50.zh \
--arpa /mnt/train.token.clean.50.lm.zh \
--discount_fallback

$MOSE_BIN/lmplz \
-o 3 \
-S 50% \
-T /mnt/tmp \
--text /mnt/train.token.clean.50.en \
--arpa /mnt/train.token.clean.50.lm.en \
--discount_fallback
```

3.1 Build Binary Language Model

The binary language model is used to save memory and speed up the decoding process. The command is shown in *Listing 5*.

Listing 5: Build Binary Language Model

```
$MOSES_BIN/build_binary \
/mnt/train.token.clean.50.lm.en \
/mnt/train.token.clean.50.blm.en

$MOSES_BIN/build_binary \
/mnt/train.token.clean.50.lm.zh \
/mnt/train.token.clean.50.blm.zh
```

3.2 Test Language Model

Listing 6: Test Language Model

```
echo "is this an English sentence ?" | \
$MOSES_BIN/query \
/mnt/train.token.clean.50.blm.en
```

Listing 7: Query Output

```
is=18 2 -2.6965108 this=45 3 -0.84844
    an=281 3 -2.317651 English=0 1
    -6.3741117 sentence=6235 1
    -4.3026004 ?=54 2 -2.2268012 </s>=2
    3 -0.21853548 Total: -18.984652
    OOV: 1
Perplexity including OOVs:
    515.3390835409365
Perplexity excluding OOVs:
    126.40278774929547
OOVs: 1
Tokens: 7
Name:query  VmPeak:62996 kB VmRSS:3864
    kB RSSMax:47872 kB user:0.004339
    sys:0.008679 CPU:0.013018
    real:0.0113772
```

4 Training