Natural Language Processing
Course Project
Pinyin-to-Character Conversion

Haoming Lu, 5140309556

June 11, 2017

Abstract

In this report, I will introduce a model which converts Pinyin to Chinese characters with recurrent neural network. The model is based on OpenNMT[1], a open-source neural machine translation system. Aside from adjusting parameters of the net, some other work such as supplementary data generation, error analysis and manual simulation has been done in order to improve the performance of the model.

1 Models

1.1 Baselines

The project offered a baseline from Google IME only with the accuracy on the dataset. To comprehensively judge the performance of models on this dataset, we implemented simulation with Sogou IME and manually labelled 1000 sentences in development dataset[2] as two extra baselines. In the following sections, results produced by these baselines will be referred to analyze the dataset and errors of the model.

1.2 Structure

The overall structure is shown in Figure 1, the process consists of several parts: prepare, preprocess, train, evaluate and vote.

- **Prepare**
  I attempted to find some supplementary data to expand the dataset. Since the original dataset comes from daily dialogue, subtitles of movies and TV series are considered as suitable sources because their distributions are more alike. In this project, subtitles of *Friends*, *Prison Break* and

[1] http://opennmt.net/OpenNMT/

[2] This work is accomplished in cooperation with Xueyuan Zhao, Yunqi Li, and Zhijian Liu.

Veep are added to the dataset. The process of addition consists of three parts: extract pure conversation character sequences, generate their phonetic notation and union them with the original dataset. Corresponding codes are available in folder *./prepare/*.

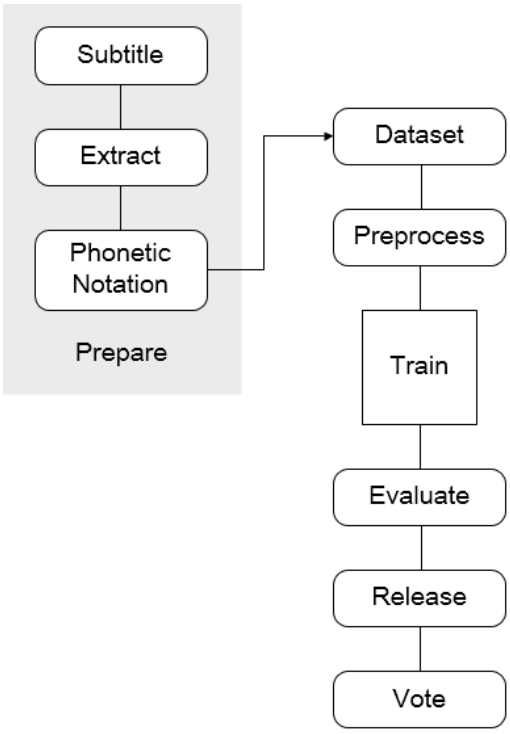

Figure 1: Structure

- **Preprocess**
  After comparing the results between models with and without word segmentation, we found that word segmentation reduces the accuracy on development set to some extent. So this method is abandoned and characters will be translated one by one.
- **Train**
  The toolkit OpenNMT offers a lot of adjustable parameters and model designs. After attempting with serveral configurations, those who have the best performances are retained.
- **Evaluate**
  All models are trained with excessive epochs, and those perform best on the development set will be released and saved. Three parts above are implemented in *./main.py*.

- **Vote**
  Eventually there will be a vote among the best-performed models, the answer that is supported by more models will be outputed when a contradiction happens. The following procedure returns a result that is suggested by strictly more than half of the candidate models if there is any, otherwise it would return an empty string. And during voting for the test dataset, no empty string was produced.

```
answer = ""
tot = 0
for i in range(cnt):
   info[i][j] = info[i][j][:-1]
   if answer == "" or answer == info[i][j]:
        answer = info[i][j]
        tot += 1
   else:
        tot -= 1
        if tot == 0:
            answer = ""
return answer
```

2 Experiment Results

The different configurations and corresponding results are shown below, in all models residual=true and dropout=0.4. We can see that all models perform only slightly different, and the translation for testdata would be the vote result of all these models.

Table 1: Configuration and results

Configuration	Validation Result
layers=2, rnnSize=256, encoderType=brnn	66.2%
layers=2, rnnSize=512, encoderType=gnmt	66.8%
layers=2, rnnSize=512, encoderType=brnn	67.1%
layers=3, rnnSize=512, encoderType=brnn	68.1%
layers=3, rnnSize=512, encoderType=gnmt	68.5%
layers=3, rnnSize=1024, encoderType=brnn	67.4%
layers=4, rnnSize=512, encoderType=brnn	68.3%

3 Analysis

In this section, we generate conversion results from manual translation, IME simulation [3], model output and ground truth of the first 1000 sentences in the development dataset, and then analyse errors of the model by comparing the results among different methods.

[3]Here we choose to simulate keyborad input with newly installed Sogou IME.

3.1 Accuracy

As is shown in Figure 2, with common sense and the help of IME, huamn translation has the highest accuracy among the three methods. The model, which has learned about features from the dataset, performs slightly worse than human. The Sogou IME without any learning process or knowledge base performs worst.
It is reasonable that manual translation has the best performance, as humans have knowledge base to translate proper nouns like personal names or place names, and can also learn the features of this specific dataset. The models, which is also able to learn the features, perform only a little worse for the lack of common sense. However, as we are going to show in next part, this dataset is extremely distinctive, so it is hard for newly installed IME to recongize uncommon words or grammer without training.

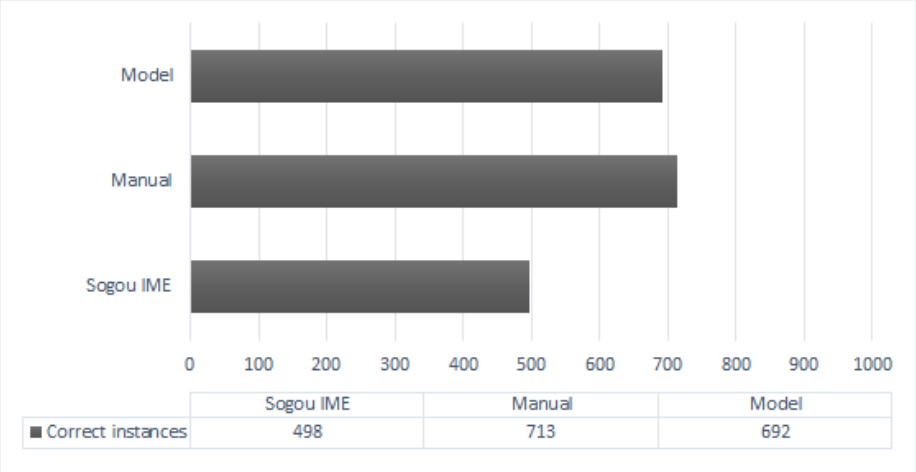

Figure 2: The number of correct instances in the first 1000 sentences from development dataset

3.2 Error Claasification

We analysed errors made by the model in the first 1000 sentences in the development dataset, they can be roughly divided into six types.

- **Normal**
  The model indeed made one or more mistakes, it might be caused by failing to recognize words that never appeared before or being unable to understand the actual meaning of sentences.

Example:
Pinyin: ni zhen lan wo bu dian nao
Ground truth: 你整烂我部电脑
Model output: 你正烂我不电脑

- **Ambiguity**
  Some characters or words with same Pinyin could also have similar usages or meaning, it is hard to distinguish them for neither humans or models.
  Example:
  Pinyin: shi shang zui qiang shou zhi tiao zhan
  Ground truth: 史上最强手指挑战
  Model output: 世上最强手指挑战

- **Wrong character**
  There could be some wrongly written characters in the dataset, here the correct outputs from the models could be judged wrong.
  Example:
  Pinyin: ming tian ji de jie wo le
  Ground truth: 明天记得借我勒
  Model output: 明天记得借我了

- **Proper noun**
  It is difficult for both humans and models to understand a personal name or place name if it has never been learned before.
  Example:
  Pinyin: san sheng san shi shi li tao hua
  Ground truth: 三生三世十里桃花
  Model output: 三生三十十里桃花

- **Meaningless**
  The original sentence has no actual meaning.
  Example:
  Pinyin: er jia xing qu you ma di
  Ground truth: 而家行去油麻地
  Model output: 而家兴趣有马地

- **Polyphone**
  The dateset provides wrong phonetic alphabets for some polyphone.
  Example:
  Pinyin: luan shi ge ge de zhuan zhang
  Ground truth: 乱是哥哥的专长
  Model output: 乱是哥哥的转账

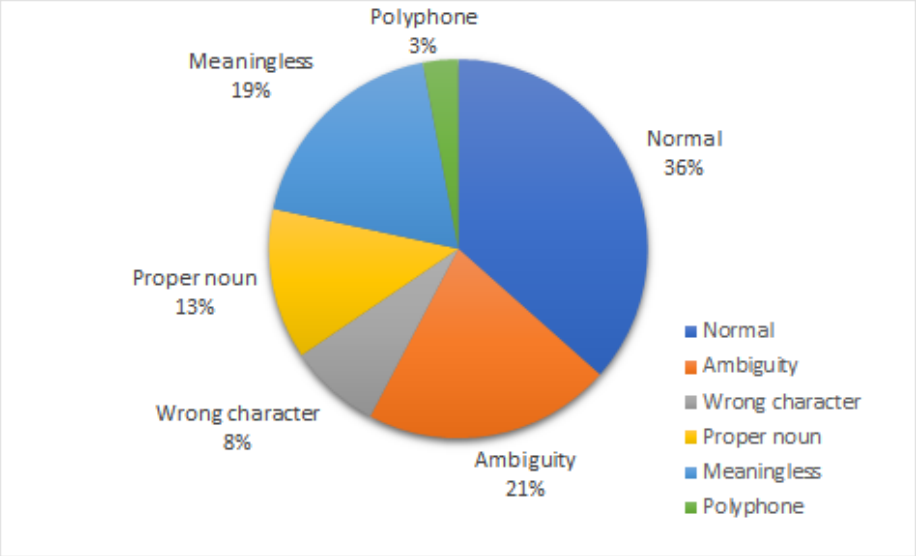

Figure 3: Distribution on different error types

Accorind to the analysis above, we could learn that only about 49% (Normal and Proper Noun) of the mistakes are evitable, while some are caused by indistinguishable modal particles such as "吗" or "嘛", and some others are caused by mistakes inside the dataset. Moreover, we noticed that there are some Cantonese instances in the dataset, which might mislead the learning process to some extent.

4 Conclusion

To obtain a more efficient model based on NMT, we may need to reduce the noise in the dataset to avoid misleading as well as expand the coverage to help the model learn proper nouns. A better-designed dataset would both increase the speed of training and imporve the accuracy of results.