Spell Correction Report

Yixiong Chen 16307110231

2019.10.17

1. Introduction

Spell correction is a typical natrual language processing task, it requires program to check whether a word is right or not and rectify wrong words efficiently. Language model and noisy channel model are two useful models to implement this function, I used both of them to implement my spell correction model.

2. Method

The spell correction model I implemented is as follows:

First, the input format of this model is sentences with their orders and number of errors. Each line contains one sample with three items, including sentence id, number of error words and the sentence. They are separated by tabs.

Second, for each sentence, the model use tokenization tool to seperate the sentence into single tokens and turn them into lowercase.

Third, because we know how many errors there are in one sentence, the model detect each word if it's a non-existing word according to the dictionary. If a word doesn't exist in the dictionary, we use the wrong word to generate several candidate correct words within 2 edit distance. Put them into the sentence, the one which has the largest possibility $P(w_{i-1}, w_i)P(w_i, w_{i+1})$ will be chosen.

Fourth, after detecting and correcting the non-existing words, if there's no error left, the sentence's correction is finishd. However, if there are still another errors, they must be real word errors. To deal with the real word errors, the model detect every word if there is any other real word w can be edit from this word x within 1 edit distance, and satisfies $P(x|w)P(w)$ larger than $P(x|x)P(x)$. According to the noisy channel model, if there is any, the word may be wrong. The model will pick "$all\ mistakes - non\,existing\,word\,mistakes$" words that have the largest value of $P(x|w)P(w) - P(x|x)P(x)$ as real word errors. With these error words, the model will repeat the procedure of correcting non-existing error to these errors, pick the ones with the largest possibility $P(w_{i-1}, w_i)P(w_i, w_{i+1})$ to be those will be chosen.

Remarks: all training data and testing data are changed to lowercase for convenience; the values of $P(x|w)$ are computed using "count_1edit.txt" according to the noisy channel model; the values of $P(w)$, $P(w_{i-1}, w_i)$ are computed using brown corpus and reuters corpus with Laplace smoothing according to language model.

3. Configurations to run the program

At the beginning of the program, the working directory should be set correctly to the directory which contains the training data such as 'vocab.txt'. Finally, the answer file 'result.txt' will also generated to this directory.

The hyperparameters m, n can be set to the numbers greater than 3. The larger they are, the better result the model will give theoretically. But bigger hyperparameters will cause longer running time.

4. Evaluation result

The evaluation result of the model with 'textdata.txt' is 90.2%. If we add more training data to train the language model and noisy channel model, it would be higher. But if it cannot reach this number in other text data, it's not strange. Because for higher accuracy, the model considers some proper nouns in 'textdata.txt', it may affect the generalization to other dataset.