Project 1 Spelling Correction

Date: February 26, 2023

1 Methodology

In this project, we implemented the Noisy channel model and language model. A combination of the two is also included. The language model computes the frequencies of words with $nltk$ corpus. The noisy channel model consists of the generation of candidates, the calculation of the probability and selection of the correct word. The mathematical expression is

$$\begin{aligned}\hat{w} &= \text{argmax}_{w \in V} P(w|x) \\ &= mboxargmax_{w \in V} \frac{P(x|w)P(w)}{P(x)} \\ &= \text{argmax}_{w \in V} P(x|w)P(w)\end{aligned}$$

where the production in the last expression can be interpreted as **edit probability** multiplied by **unigram probability**.

1.1 How to get candidates

There artwo situations where words are misspelled, typographic errors and orthographic errors. The former refers to words with similar spelling, and the latter refers to words with similar pronunciation. In this project we only consider typographic errors.

1.1.1 Run through dictionary

One way to find similar spelling words is to enumerate every word in the dictionary and compute the edit distance for every two, which will take lots of time once the vocabulary is large.

1.1.2 Generate (less than) k-edit distance words

Another way can be generating all words within edit distance k (in this implementation, $k = 2$), and then checking whether these words are in the dictionary, which is the method we use.

1.2 How to get channel model

1.2.1 Generate confusion matrix

Here we use the corpus from Norvig's $spell-erroe.txt$ on the website "Natural Language Corpus Data: Beautiful Data" to manually generate the confusion matrix.

1.2.2 Compute the probability

we use the definition of the **edit probability** and confusion matrix to compute

$$P(x|w) = \begin{cases} \frac{\text{del}[w_{i-1},w_i]}{\text{count}[w_{i-1}w_i]}, \text{ if deletion} \\ \frac{\text{ins}[w_{i-1},x_i]}{\text{count}[w_{i-1}]}, \quad \text{if insertion} \\ \frac{\text{sub}[x_i,w_i]}{\text{count}[w_i]}, \quad \text{if substitution} \\ \frac{\text{trans}[w_i,w_{i+1}]}{\text{count}[w_iw_{i+1}]}, \text{ if transposition} \end{cases}$$

1.2.3 Combination comparison with language model

To show the difference between these two models, we tried them separately. However it's surprising that the combination of these two does a better job. The combination is simply expressed as

$$\log(P(w_{i-1}, w_i) * P(w_i, w_{i+1})) + \log(P_{edit})$$

2 Experiments and Results

2.1 Training N-gram model

To get an intuition of the influence of corpus on the N-gram model, we fixed the bigram model and tried 3 different corpora, $ans.txt$, $nltk\ reuters$ and $nltk\ brown$. The results are as follow:

Table 1: Influence of corpus

corpus	Accuracy
ans.txt	86.9%
reuters	86.7%
brown	82.2%

Take the generalization of the model, the $nltk\ reuters$ would be a great corpus to use.

2.2 Smooth N-gram model

There are lots of discussions and experiments on the different methods of smoothing. Here we choose interpolation smoothing to deal with the limitation of samples. The expression looks like

$$\begin{aligned} p(w_n|w_{n-1}w_{n-2}) &= \lambda_1 p(w_n|w_{n-1}w_{n-2}) \\ &+ \lambda_2 p(w_n|w_{n-1}) \\ &+ \lambda_3 p(w_n) \end{aligned}$$

$$\sum_i \lambda_i = 1$$

I've tried several experiments to discover the relation between λ and $Accuracy$, and get results as follow:

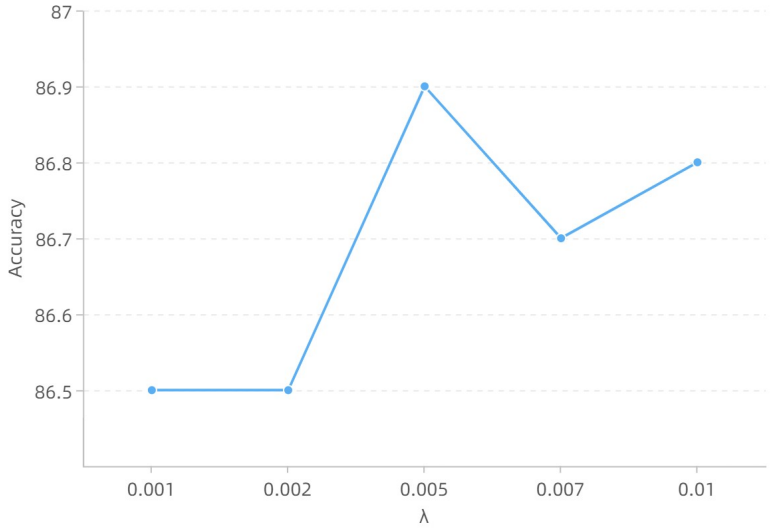

Figure 1: $Accuracy$-λ chart

The reason why the accuracy of the specific value of lambda is higher than either larger or smaller ones may be that there are not many zeros in the corpus and we have to find the balance. Here $\lambda = 0.005$ has the best performance.

2.3 Language Channel model

The table below summarizes the characteristics between language models and channel models, while also giving us some insight into their combination.

Here, $channel$ refers to typical channel model, $channel(bigram)$ refers to channel model with bigram probability rather than unigram, $channel(2bigram)$ represents the channel model with double way bigram

Table 2: Comparison between language model and channel model

Model	Accuracy
channel	84.9%
channel(bigram)	86.1%
channel(2bigram)	**86.9**%
channel(1trigram)	75.3%
channel(3trigram)	76.2%
unigram	84%
bigram	86.3%

model, $channel(1trigram)$ means the channel model with trivial trigram, and $channel(3trigram)$ stands for channel model multiplied with typically implemented trigram model. Obviously, the channel model with double-way bigram performs the best. The relatively low accuracy with trigram might be due to a small probability after multiplying more than 3 times, possibly getting a probability inaccurately.

3 Reflection and Summary

3.1 Learning Process

At first, I just read the PPT provided by TAs and found some concepts unclear. Then thanks to one of our teammates, through the video he suggested, I overcome that unclearness.

When it comes to implementing my ideas, I found it useful to browse some related website and tutorials, like Norvig's "How to Write a Spelling Corrector".

More importantly, the discussion between one of our teammates and me really helps me a lot, raising my accuracy by about 2% after fixing the case issues. I believe more discussions will bring more academic help.

3.2 Way to implement

Spelling correction is nothing a small task. Fortunately, we chose to dive this huge task into small pieces, like confusion matrix generation, language model training, data preprocessing, etc. The concept of divide and conquer leaves great impression on me, and it helps me get through the project.

3.3 What I get from it

Even though I'm currently out of China, I felt really tight to other teammates through the discussions about this project. We share our ideas, which broadens my eyes and gives me more inspiration. Lucky to be a part of the team.