Spell Corrector Doc

ML_EPFL Spell Corrector

Abstract

I was given mis-spelled query, our goal was to choose the “nearest” one of various alternative correct spellings. Concerning this problem, we need a notion of nearness. I defined deletion (remove one letter), a transposition (swap adjacent letters), an alteration (change one letter to another) or an insertion (add a letter) as distance 1. And during the definition, I thought distance 1 and 2 are the most common type errors. Then when two correctly spelled queries are tied, I select the one that is more common. For instance, grunt and grant both seem equally plausible as corrections for grnt. Then, the algorithm should choose the more common of grunt and grant as the correction. The idea is to use the correction that is most common among queries typed in by other users. The idea here is that if grunt is typed as a query more often than grant, then it is more likely that the user who typed grnt intended to type the query grunt. Having these ideas in mind, I used Bayesian Theorem to split the problem into two parts, which are language model and error model. Finally, I used The Complete Work of William Shakespeare to do the language model, getting right correctness for error words.

Bayesian Theorem

- Assume given the “wrong” word w, our goal is to find the “correct” work c, i.e. we want to get $argmax_c P(c|w)$.
- It is equivalent to $argmax_c P(w|c)P(c)/P(w)$, according to the Bayesian Theorem.
- $P(w)$ is the same for every possible c, we can ignore it, thus we have: $argmax_c P(w|c)P(c)$.

Model Theory

- **Language Model**: The value of $P(c)$ is the probability of the correct[1] word. Usually it can be got from a language model, i.e. it is the frequency of the assuming correct work in a big English text. [2] In my implementation, I use the Complete Works of William Shakespeare[3] as the frequency counter.
- **Smoothing in Language Model**: Treat novel words as if we had seen them once.
- **Error Model**: The value of $P(w|c)$ can be got from a error model, i.e., usually it can be solved by edit distance.

Why do we use Bayesian Theorem?

- Estimating $P(c|w)$ we have to consider both the probability of c and the probability of the change from c to w anyway. So it is easier to separate the model into two models, thus we can handle one model at a time.

TODO

- This is only a simple word spell checker and corrector. Although I implement a words corrector function which can correct a word sequence, it is based on the word error, not the sentence error. We can use linguistic model to do the sequence corrector.

---

1. When I use correct work, usually it is referred to the assuming correct work. ↩
2. Detail of Language Model ↩
3. URL of The Complete Works of William Shakespeare ↩