BBM 497 NLP LAB. ASSIGNMENT 1

Tolgahan Dikmen
21327929
`tolgahandikmen95@gmail.com`

March 21, 2019

1 Introduction

Language Modeling is one of the fundamental concepts of Natural Language Processing (NLP). In this assignment, you will build some of the basic language models and use your language models to perform authorship-determination on a list of documents.

The problem consists three differenet parts. First part starts building own basic language unigram, bigram and trigram models which are according to given training files. Second part is generating sentences according to builded bigram and trigram models for each user. Third part is calcalulating perplexity to given test file and deciding to which author is wrote that file.

2 Part Of The Programs

2.1 First Part: Building Basic Language Models

In this task, I need to parse and tag the given file. While parsing, I added '<s>' tag for starting of the sentences and replaced '.' , '?', '!' with '</s>' tag for ending of the sentences. After sentence boundry tagging, all the punctuation marks are removed from the remaining data. After the punctuation marks were removed, I switched to creating the models.

For building unigram models for each author, dictionary variable type is defined and filled it up with unieque tags. If tag is in the dictionary, the value is increased by 1, otherwise a new key added for that tag with value 1.

For building bigram models, program looks at the word/tag pairs. Again if tag is in the dictionary, the value is increased by 1, otherwise a new key added for that tag with value 1.

For building trigram models, program looks at the triple word/tag pairs. Again if tag is in the dictionary, the value is increased by 1, otherwise a new key added for that tag with value 1.

2.2 Second Part: Generating Sentence

I generated three sentence for each user. That sentences are written by unigram language model, bigram language model and trigram language model. After generation, porbability of the sentences is calculated.

Generated starts with sentence start boundry '<s>' for bigram language model. I searched inside of '<s>' bigram model dictionary keys. I splitted the key and if first word of splitted key equals to the '<s>', I added a new array the key and value seperately. These two different array for keys and values. I added keys directly to the array. But when I was adding the values of the keys, I used to cumulative array. Becuse after checking all keys in dictionary, I need to select one of them randomly. For this selection I generated random number and I picked the vaule from the vaules array according to generated random number. After selection is done, these steps followed one another untill 30 words or reached end of sentence boundry tag '</s>'. After first step, the other steps will be the same with each other.

The sentence generated then I calculated the probability of the sentece according to this formula;

Unigram:

$$P(w_1 w_2 w_3 ... w_n) = \prod_{i}^{n} P(w_i) \tag{1}$$

Bigram:

$$P(w_1 w_2 w_3 ... w_n) = \prod_{i}^{n} P(w_i | w_{i-1}) \tag{2}$$

Trigram:

$$P(w_1 w_2 w_3 ... w_n) = \prod_{i}^{n} P(w_i | w_{i-1} w_{i-2}) \tag{3}$$

2.3 Third Part: Finding Who Wrote This

In this part, the program have to find out who wrote the given test files. To do this, the program needs to use builded unigram, bigram and trigram language models which are created in first part. But a problem may be encountered at this point. The problem is a word can come from test file which is never used in training file and the program does not know that word information. So the calculating operations is executing, the unknown word/tag will not be resolved for the algorithm. To avoid this situaton add-one(Laplace) smoothing is applied for the out-of-vocabulary problem. The formula is as follows.

$$P(w_i | w_{i-1}) = \frac{C(w_i | w_{i-1}) + 1}{C(w_{i-1}) + V} \tag{4}$$

After handled out-of-vocabulary problem, calculated the probability and perplexity. Actually one of them is enough for deciding the author of the test file but I calculated both of them. According to the results, the program can find correctly bigram models to decide the author but trigram model sometimes. If result perplexity is lower than the other authors' result, then the test file should write to the test file.

3 References

https://www.w3schools.com/python/python_dictionaries.asp
https://web.cs.hacettepe.edu.tr/~burcucan/BBM495.htm
https://www.tutorialspoint.com/python/python_functions.htm
https://nlp.stanford.edu/~wcmac/papers/20050421-smoothing-tutorial.pdf
https://en.wikipedia.org/wiki/Additive_smoothing