NLP Task 1- Vanilla Language Model

Deborah Vella

Corpus

I chose a Maltese corpus called malti03.academic.1.txt. For this corpus, I took the first word of every line in the text file and stored it inside a list. Therefore, each element of the list contains tokens which are words or punctuation. To read this file I made use of the utf-8 decoding so that the Maltese letters are not changed into random symbols.

Storage

Lists and arrays

As mentioned above, I stored the words of the corpus inside a list, one after the other. This list was then used to get all sentences and created a 2D array having each row being a sentence and each column storing words. This was then shuffled randomly and split it 80/20. I used the training data to calculate the N-Grams upon.

An array for each N-gram model was created to store the unique permutations found. I also created arrays to store the frequencies of the unigram and the bigram models. These arrays are populated every time a new unique N-gram is met. E.g the word "għalliema" is stored at index 0 in the unigram array and its respective frequency is stored at index 0 in the unigram's separate frequency array.

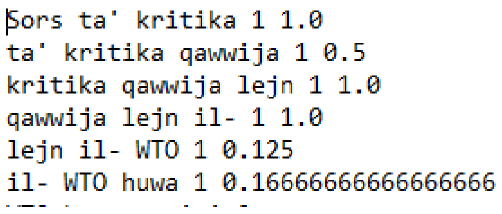
By storing the frequencies, the program can make use of them when calculating the probabilities later on, instead of calculating the frequency again. E.g to calculate the probability of a trigram, for the denominator it accesses the bigram's frequency which would have been previously calculated.

Text Files

For each N-Gram model i created a text file to store the details in. Once again the utf-8 encoding/decoding was used, this time to write to the files.

The first column consists of unique unigrams/bigrams/trigram depending which one of them is being analyzed. The second column consits of the frequency of the N-Gram, and the last column is the probability. These are separated with a space character. I chose this character because no token contains a space characters, therefore, it is not confused with an actual token (e.g. if the delimiter is a comma it can be confused with a token from the corpus).

The below is a snippet from the trigram text file



```
Sors ta' kritika 1 1.0
ta' kritika qawwija 1 0.5
kritika qawwija lejn 1 1.0
qawwija lejn il- 1 1.0
lejn il- WTO 1 0.125
il- WTO huwa 1 0.16666666666666666
```

Formulae used

I used the below formula to calculate the probabilities of the bigrams. In the case of the trigram I edited the formula to suite the trigram model. So, the denominator is the count of the 2 previous words. And the numerator is the frequency of the three words.

$$P(w_i \mid w_{i-1}) = \frac{count(w_{i-1}, w_i)}{count(w_{i-1})}$$

Time to build

It took the program about 23 seconds to finish running when using my laptop.

Running the program

You can double click on the RunMe.cmd file to run the program. When it finishes you can press enter to close the window. All the files for running the program are found inside the Assignment NLP folder. The text files contain all the results I got when I ran the program.