IT412: Natural Language Processing

Assignment 6: N-gram Language Model

Instructor: Prasenjit Majumder

1 Language Model

Language Modeling (LM) is one of the most important parts of modern Natural Language Processing (NLP). Language Modeling is the task of predicting what word comes next. Language model is required to represent the text to a form understandable from the machine point of view. It finds application in various NLP tasks i.e Machine Translation, Spell Correction, Summarization etc.
Given a sequence of words $(w^{(1)}, w^{(2)}, ..., w^{(t)})$, probability distribution for next word $w^{(t+1)}$ can be computed as

$$P(w^{(1)}, w^{(2)}, ..., w^{(t)}, w^{(t+1)}) = P(w^{(t+1)} \mid w^{(1)}, w^{(2)}, ..., w^{(t)})$$

where $w^{(t+1)}$ can be any word from vocabulary $V = \{w_1, ..., w_{|V|}\}$

To estimate each probability a straightforward solution could be to use simple counting.

$$P(w^{(t+1)} \mid w^{(t)}, ..., w^{(1)}) = \frac{count(w^{(t+1)}, ..., w^{(1)})}{count(w^{(t)}, ..., w^{(1)})}$$

1.1 Approaches to Language Modelling

1. **N-gram model A Statistical approach**
   An N-gram is a contiguous (order matters) sequence of items, which in this case is the words in text.It is based on Markov's assumption, that $w^{(t+1)}$ depends on the preceding $n-1$ words.

$$P(w^{(t+1)} \mid w^{(t)}, ..., w^{(1)}) = P(w^{(t+1)} \mid w^{(t)}, ..., w^{(t-n+1)})$$
$$P(w^{(t+1)} \mid w^{(t)}, ..., w^{(t-n+1)}) \approx \frac{count(w^{(t+1)}, ..., w^{(t-n+1)})}{count(w^{(t)}, ..., w^{(t-n+1)})}$$

   For a sequence of word i.e a sentence of length n, bi-gram model can be computed as

$$P(w^n) \approx \prod_{k=1}^{N} P(w_k|w_{k-1})$$

1.2 Language Model evaluation – Perplexity

The good LM should calculate higher probabilities to "real" and "frequently observed" sentences than the ones that are wrong accordingly to natural language grammar or those that are rarely observed.

Perplexity metric is the probability of the test set, normalized for the length of the probability by the number of words. Perplexity (PPL) of a model over a sequence $W = [w_1, ..., w_N]$ is calculated as

$$PPL = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_1, ..., w_{i-1})}}$$

Thus, if we are computing the perplexity of W with a bi-gram language model, we get:

$$PPL(W) = \sqrt[N]{\prod_{i=1}^{N} \frac{1}{P(w_i \mid w_{i-1})}}$$

Lower the perplexity (PPL) better the Language Model.

2 Implementation

2.1 Dataset

You are provided with dataset that contains four folders, out of which use folder 2004,2005 and 2006 for training and 2007 for testing. Each folder has UTF files from which you need to extract Text part.

2.2 Implementation Exercise

1. Create Uni-gram, Bi-gram and Tri-gram Language model for given dataset and compute perplexity for them.

2. Generate random sentences by models you have trained by giving starting words. For example give starting words of sentence "today the ______ ..."

3 References

- `https://web.stanford.edu/~jurafsky/slp3/3.pdf`

3.1 Codes

- `https://github.com/ollie283/language-models`
- `https://www.analyticsvidhya.com/blog/2019/08/comprehensive-guide-language-model-nlp-python-code/`

3.2 Datasets

List of few publicly available word level datasets are listed down.

- Penn Treebank
- Google Billion Word benchmark
- Gutenberg Dataset
- WikiText