E1246 - Natural Lnaguage Understanding
Assignment1 : Language Models

Tapan Bhardwaj(14704)
tapanb@iisc.ac.in

Abstract

To designed a language model on **Brown**(D1) corpus and **Gutenberg**(D2) corpus using ngrams. Following are two tasks that we performed -

- **Task 1:** Divide both dataset into train, dev, and test and build the best LM in the following settings and evaluate.
  - **S1**: Train: D1-Train, Test: D1-Test
  - **S2**: Train: D2-Train, Test: D2-Test
  - **S3**: Train: D1-Train + D2-Train, Test: D1-Test
  - **S4**: Train: D1-Train + D2-Train, Test: D2-Test
- **Task 2:** Generate few sentences of 10 tokens.

1 Implementation

1.1 Language Model

I have implemented four language models for given datasets. Initially I have added ”UNK” word to training corpus to make it closed vocabulary.

- First model is simple unigram back-off model. In this model we look for probability of unigram and if unigram is absent in training corpus we assign unigram probability of ”UNK” word.
- Second model is simple bigram back-off model.In this model we look for probability of bigram and if it is absent we look for probability of unigram and if unigram is absent in train we assign unigram probability of ”UNK” word.
- Third model is simple trigram back-off model.In this model we look for probability of trigram and if it is absent we look for probability of bi-gram and if it is absent then it look for the probability of it’s uni-gram and if unigram is absent in train we assign unigram probability of ”UNK” word.
- Fourth model is simple fourgram back-off model. In this model we look for probability of fourgram and if it is absent we do recurcively like above.

1.2 Random Sentence Generation

For sentence generation i have used four-gram model. I have assumed three starting symbol(*) to be present and then choosing a word according to four-gram probability distribution from the list of all words that are possible(i.e. words that follow these two words in training set) and repeating the same process to generate the token of required length.

2 Result

2.1 Task 1

For Simple unigram-backoff - Train: 90per Test: 10per
For Simple bigram-backoff - Train: 90per Test: 10per
For Simple trigram-backoff - Train: 90per Test: 10per
For Simple fourgram-backoff - Train: 90per Test: 10per

S1 : train = D1 train and test = D1 test	
Simple uni-gram backoff	1308.280
Simple bi-gram backoff	330.399
Simple tri-gram backoff	249.650
Simple four-gram backoff	237.871
S2 : train = D2 train and test = D2 test	
Simple uni-gram backoff	753.285
Simple bi-gram backoff	155.347
Simple tri-gram backoff	97.505
Simple four-gram backoff	83.415
S3 : train = D1 train+D2 train and test = D1 test	
Simple uni-gram backoff	1639.992
Simple bi-gram backoff	420.491
Simple tri-gram backoff	307.129
Simple four-gram backoff	287.738
S4 : train = D1 train+D2 train and test = D2 test	
Simple uni-gram backoff	789.880
Simple bi-gram backoff	157.184
Simple tri-gram backoff	96.357
Simple four-gram backoff	81.269

2.2 Task 2

Some examples of token generated from Language Model:

- The funeral for my husband was just what Letch deserved very
- The operations of its other plant in Rochdale and Leesona's
- It is recognized that all the obviously non-propagandistic aspects of
- Outside the hall I anxiously looked around for the bubbles

3 Acuuracy/Measures

3.1 Task 1

Perpexility is used as the measure for this task. (All values in Result section table contain perpexility value)

3.2 Task 2

Human Evaluation.

4 Git Hub Link

https://github.com/TapanBhardwaj/NLP-projects/