EROL ÖZKAN

N18245609

erolozkan@outlook.com

CMP 670

Statistical Natural Language Processing (Spring 2019) Homework 1

# BASIC MODEL

## Data Cleaning

Beginning and end of sentence speech tags (<s> and </s>) are added to all sentences. An example is shown in Table 1.

Table : Data Cleaning Example

|  |
| --- |
| I am Sam → <s> I am Sam </s> |

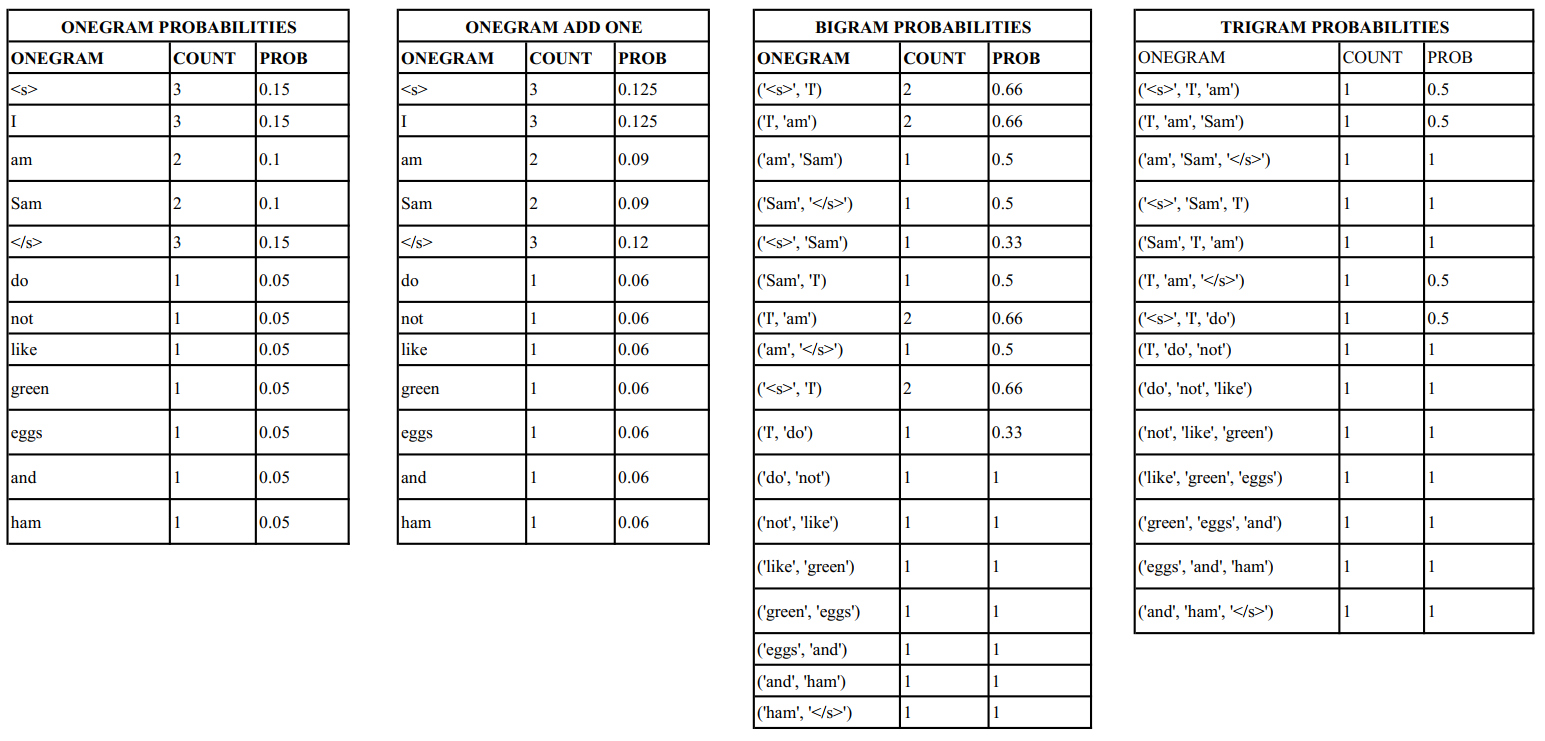
## N-Gram Language Model – Simple Test (Runner\_First.py)

3 simple sentences are given in Table 2. From these sentences Unigram, Bigram and Trigram models are trained. Some of the statistics are shown in Table 3.

Table : Example Training Corpus

|  |
| --- |
| <s> I am Sam </s> |
| <s> Sam I am </s> |
| <s> I do not like green eggs and ham </s> |

Table : Some Examples from trained N-gram model



## Laplace Smoothing (output/1gram-add-one.txt)

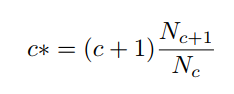
For unigram model Laplace Smoothing is implemented. Table 3 shows Laplace Smoothing results. Here K value is 1. So it is simply Add-one Smoothing. Formula is shown below.

## Good-Turing Smoothing (output/good\_turing\_smooting\*.txt)

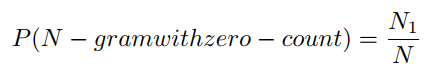
For Bigram and Trigram model Good-Turing Smoothing is implemented. For this purpose frequencies of frequencies are calculated using below formula.



Using this c values c\* values are calculated with below formula. These new c∗ values are then used to replace the maximum likelihood scores. Note that there exists some *Nc*+1 values which are zero. Therefore, some c\*values for these values are also zero. This is because of data sparsity problem. There are some methods to estimate this zero values.



The purpose of Good-Turing Smoothing is to estimate the frequency of zero count events. To do this bigram zero occurrence and trigram zero occurrence probabilities are calculated using below formula. In this equation, *N*1 is the number of counts seen once and N is the total number of counts seen in the training corpus. These values are assigned to test dataset words that never occurred in the training corpus.



## N-Gram Language Model – Real Dataset Test - Perplexity of Test Dataset

Perplexity of test data is calculated using unigram, bigram and trigram models. Results are shown in Table 4.

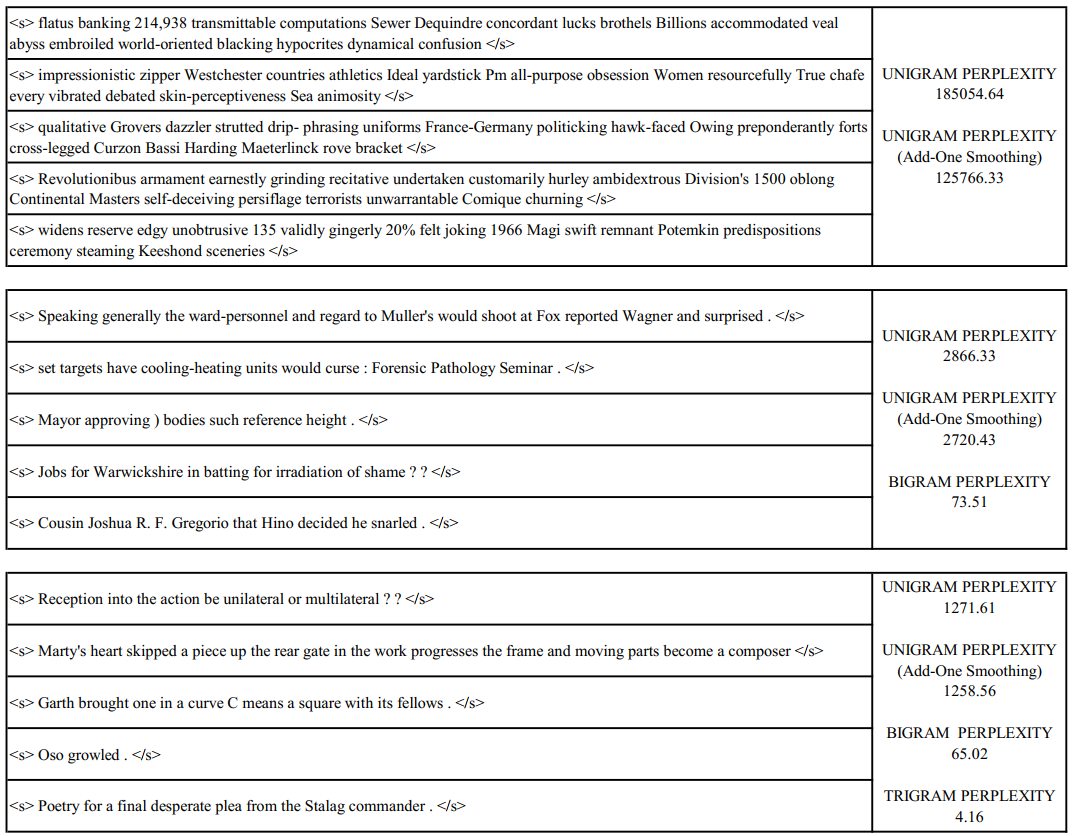
Table : Perplexity Scores of Test Data

|  |  |
| --- | --- |
| **UNIGRAM PERPLEXITY** | 726.57 add-one smoothing used |
| **BIGRAM PERPLEXITY** | 17.52 log2 (0) is assumed as zero. Some c\* values are zero. |
| **TRIGRAM PERPLEXITY** | 1.97 log2 (0) is assumed as zero. Some c\* values are zero. |

## N-Gram Language Model – Generate Sentences, Calculate Perplexity

Foreach unigram, bigram and trigram model, 5 sentences are generated. Word are selected randomly from the corpus. That’s why, for example, the probability of p(“</s>”), p(“.”) or p(“</s>” \ “.”) are high. Perplexity of these sentences are calculated. Generated sentences and their perplexity score are given in Table 5. log2 (0) is assumed as zero not as -infinite.

Table : Randomly Genereated Sentences and Their Perplexity Score



* Unigram model selects random words without using any context information for the words. That’s why, generated sentences have no meanings and the grammar is the worst.
* Bigram model selects the first word as “<s>”, then randomly generates next words using bigram model ( p(x\<s>) ). It is better than the unigram model and sentences are grammatically more correct. Perplexity is also way lower.
* Trigram model selects first word as “<s>”. It selects second word using bigram model, then generates next words using trigram model. Sentences are better than both unigram and bigram model considering both meaning and the grammar.

## Error Analysis

* There are unigram, bigram and trigram word units that occur in test dataset, but does not occur in training dataset. The probability of these word unit’s is 0. log2(0)=-infinite without any smoothing. In order to calculate Perplexity, I considered these values as zero! Only add-one smoothing has non-zero values. There were zero values in Good-Turing Smoothing also.
* As going from unigram to trigram, the probability of test dataset word unit does not occur in training dataset increases. That’s why in these cases, some interpolation or back-off strategies are necessary.

## Language Model Evaluation

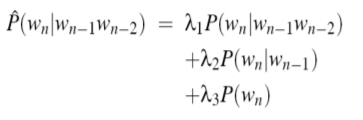
**Entropy:** a measure of uncertainty/disorder

**Cross-entropy**: Model should have low uncertainty (entropy) about which word comes next. (Lower cross-entropy ⇒ model is better at predicting next word.)

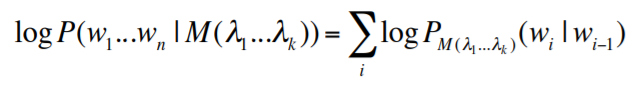
**Perplexity:** 2cross-entropy Lower perplexity is better for a language model

# LINEAR INTERPOLATION

Simple interpolation formula is given below.

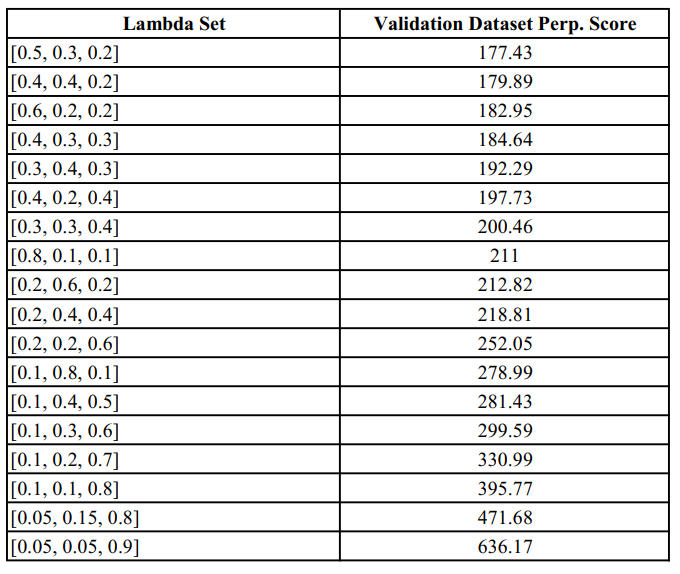


We are trying to find λs to maximize the probability of held-out data. This formulate is shown below.



One of the simplest technique is to build an interpolated language model using brute-force approach. Note that sum of the λs equals to one. Perplexity scores using different lambda values are shown in Table 6.

Table : Validation Dataset Perplexity Scores Using Different Lambda Values



Test dataset best perplexity score using lambda set **[Unigram=0.5, Bigram=0.3, Trigram=0.2]** is 177.43.

# DISCOUNTING(output/\*discounted\*.txt)

Below table shows the discounted probabilities with β=0.5 and saved probability mass.

