Homework 1

Names: Loyd Flores
Department: Computer Science

Queens College
New York, NY

October 15th, 2024

Abstract

This homework focuses on prepossessing techniques, modeling with n-Grams, and smoothing techniques.

Contents

1 Section 1: Pre-processing

1.1 Padding Sentences

Language models learn through processing large amounts of text data which is often referred to as *Corpus*. In this assignment there exists two corpora. The *Training corpus* will be used to train the models that will be discussed in later sections. This allows models to estimate the probabilities of unseen text, or in the context of this homework *Testing corpus*, based off the information it can obtain in the training corpus. The larger the training corpus is the better it can estimate probabilities on unseen text. Both the training and testing corpus were extracted from the NewsCrawl corpus. *train.txt* contains about 100,000 sentences and the testing corpus is about 120 sentences.

The first step of pre-processing is iteratively padding the sentences with the start (<**unk**>) token and the end token (<**unk**>). This allows the models to gain more context by understanding the where a sentence starts and ends.

Listing 1: Before Padding

```
1: Under the trust 's Fit for the Future proposals , two of the hospitals would lose
    their A&E units with consultant - led maternity services also possibly being
    centralised on one site .

2: Man charged over drugs seizure

3: Borrowing from your 401 ( k ) or withdrawing from your IRA , on the other hand ,
    can make sense in some situations , says Michael Chasnoff , a financial planner in
    Cincinnati : " If going back to school helps you command a higher level of income
    , there is a return on your investment . "
```

Listing 2: After Padding

```
1: <s> Under the trust 's Fit for the Future proposals , two of the hospitals would
    lose their A&E units with consultant - led maternity services also possibly being
    centralised on one site . </s>

2: <s> Man charged over drugs seizure </s>

3: <s> Borrowing from your 401 ( k ) or withdrawing from your IRA , on the other hand
    , can make sense in some situations , says Michael Chasnoff , a financial planner
    in Cincinnati : " If going back to school helps you command a higher level of
    income , there is a return on your investment . " </s>
```

1.2 Lower casing and tokenization

Lowering tokens normalizes tokens for example 'Apple' is semantically the same as 'apple'. It ensures that variations of the word is treated the same. Lowering also reduces Data Sparsity to allow the model to learn more efficiently. For those reasons it is best to make sure all tokens are lower case to allow the model to generalize better.

Listing 3: After Lowercase

```
1: <s> under the trust 's fit for the future proposals , two of the hospitals would
    lose their a&e units with consultant - led maternity services also possibly being
    centralised on one site . </s>

2: <s> man charged over drugs seizure </s>

3: <s> borrowing from your 401 ( k ) or withdrawing from your ira , on the other hand
    , can make sense in some situations , says michael chasnoff , a financial planner
    in cincinnati : " if going back to school helps you command a higher level of
    income , there is a return on your investment . " </s>
```

1.3 <unk> Token

Replacing tokens that appear once or less with the <unk> token allows us to calculate probabilities more effectively. Tokens that appear only once are not significant enough, and this approach also prevents the model from encountering tokens it has never seen. If the model assigns a zero probability to an unseen word, it would result in the entire sentence or token having a probability of zero. By replacing these cases with the <unk> token, we redistribute some probability mass to those unseen tokens.

2 Section 2: Modeling

2.1 Unigram MLE

The Unigram Maximum Likelihood model calculates the probability of a token by assuming that all tokens are independent of each other which allows us to simply obtain the probability of a token from counting how many times it appeared on the training corpus. To predict a sentence, the probabilities of each token are obtained and chained together to obtain a joint probability. Training the a Unigram MLE model is assigning a probability to all tokens in the training corpus. Below shows the formula to compute the probability of a single token.

$$P(w) = \frac{\text{count of } w}{\text{total number of words in the corpus}}$$

2.2 Bigram MLE

The Bigram Maximum Likelihood model differs from the unigram because it assumes that words have some relation to the previous words, which is called a *bigram*. A bigram is pair of tokens that is composed of the current token w_i and the previous token w_{i-1}. To calculate the probability of a sentence using the Bigram MLE model, the probabilities of each bigram must be found and chained together to obtain the probability of the entire sentence. Training the a Bigram MLE model is assigning a probability to all bigrams in the training corpus. Below is the formula in obtaining a single bigram.

$$P(w_2 \mid w_1) = \frac{\text{count of } (w_1, w_2)}{\text{count of } w_1}$$

2.3 Bigram MLE with plus one smoothing

Smooothing is necessary because as previously mentioned it acccounts for tokens that did not appear in the training corpus or appeared only once. This redistributes the probability mass and allows the model to generalize better. Training the a Bigram

MLE model with plus one smoothing is assigning a probability to all bigrams while applying smoothing. To obtain the probability of a token using the Bigram MLE with plus one smoothing is:

$$P(w_2 \mid w_1) = \frac{\text{count of } (w_1, w_2) + 1}{\text{count of } w_1 + V}$$

3 Section 3: Questions

3.1 Preprocessing and Modelling

1. There were 83,045 tokens in the Vocabulary of the training corpus, subtracting 2 to disregard start and end token will result to **83,043** tokens.

2. There were a total of 100,000 start tokens we subtract that from the total number of tokens to end up with **2,468,210** tokens.

3. There are 494 tokens found in test corpus that were not in the training corpus resulting in a proportion of **18.5%** tokens that did not appear in training

4. There are 742294 bigrams in train data. There are 2390 bigrams in test data.There are 657 bigrams that appeared in test only. The proportion of bigrams found in test only is: **27%**

5. **Unigram Log Probability**: = log(P("I")) + log(P("look")) + log(P("forward")) + log(P("to")) + log(P("your")) + log(P("reply")) = **-90.7**

   Bigram Log Proabaility: log(P("I | <s>")) + log(P("look | I")) + log(P("forward | look")) + log(P("to | forward")) + log(P("your | to")) + log(P("reply | your")) + log(P("</s> | reply")) = **inf**; In code it will give a math domain error.

   Bigram Log Probability With Smoothing: The formula is the same the only difference are the estimates. Unlike the regular Bigram Log Probability, smoothing allows us to obtain the log probability which was **-34.10**

3.2 Perplexity

6. Preplexity of the text using the different models:
   Unigram: 33033.78
   Bigram MLE: inf; math domain error
   Bigram MLE with smoothing: math domain error

7. Preplexity of the train corpus using the different models:
   Unigram: 6.34
   Bigram MLE: inf; math domain error
   Bigram MLE with smoothing: math domain error