Assignment 3: Language Modeling

Nidhi Dhamnani
A59012902
CSE 256: Statistical NLP (Spring 2022)
E-mail: ndhamnani@ucsd.edu

***Abstract* – The goal of this assignment is to build a probabilistic language model and analyze different domains using the language model.**

***Keywords* – Probablistic Language Modeling, n-grams, Context Aware Modeling**

I. Dataset

The dataset consists of three corpora, namely, Brown, Reuters, and Gutenberg. The Brown corpus consists of standard American english text from 1979. The Reuters corpus consists of financial news articles that appeared on the Reuters newswire in 1987. The Gutenberg corpus consists of public domain works by authors including Jane Austen and William Shakespeare.

II. Unigram Language Model Analysis

A. *Analysis on In-Domain Text*

Perplexity is a measurement of how well a probability distribution or probability model predicts a sample. It is based on likelihood of word sequences. Since the probability of word varies highly from corpus to corpus of different vocabulary sizes, the likelihood of the of the word sequences is also very sensitive on the corpus. Formally, perplexity is defined as:

$$PP(W) = 2^{Entropy(W)} = 2^{-1/N*(log_2(P(w_1,w_2,...,w_n)))}$$

In this formula, perplexity is dependent on both the size of vocabulary as well as the probability of words.

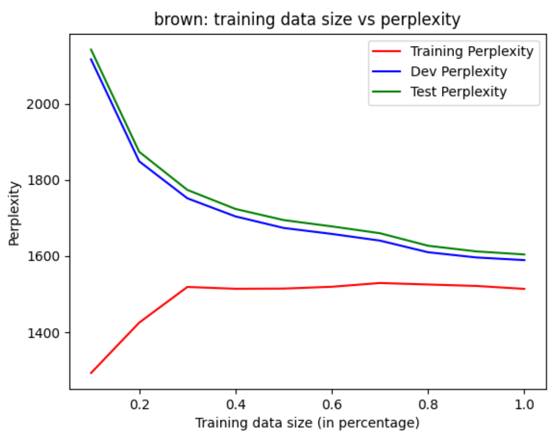

Fig. 1. Brown dataset training perplexity vs training data size

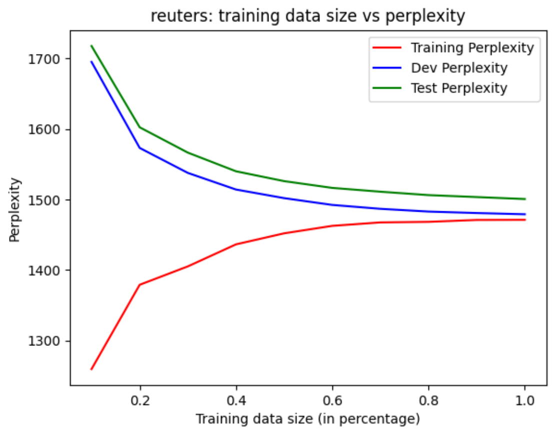

Fig. 2. Reuters dataset training perplexity vs training data size

A low perplexity indicates the probability distribution is good at predicting the sample. It can be interpreted as the weighted branching factor. For example, if we have a perplexity of x, it means that whenever the model is trying to guess the next word it is as confused as if it had to pick between x words.

As shown in Fig. 1, Fig. 2, and Fig. 3, the Test and Dev perplexity for all the three datasets decreases as the data size increases. The reason for this can be as the size of the dataset increases, the probability of the words increase, keeping the probability of $< UNK >$ terms as almost the same. Therefore, even though N increases, the overall entropy decrease and hence the perplexity decreases. In other words, as the keep on adding more words, the model is becoming more confident about the probability of words, hence the weights of the branching factor increase, therefore, the chances of making an error decreases.

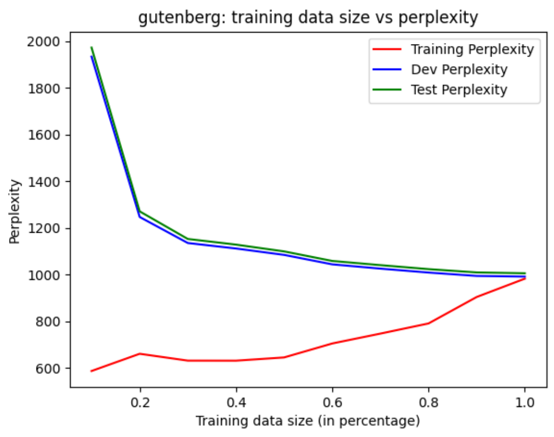

Fig. 3. Gutenberg dataset training perplexity vs training data size

The graphs also shows that the model is not overfitting on the training data otherwise the test and dev perplexity would have increased after a certain threshold instead of decreasing.

B. Analysis on Out-of-Domain Text

From Table I, Table II, and Table III, we can see that the model trained and tested against itself performs best compared to the model trained on one dataset and tested against another. We can also observe that the perplexity for model trained on Gutenberg dataset and tested against Reuters dataset is the highest on all the three - Train, Test, and Dev datasets. The reason behind this can be the mismatch in the vocabulary. Gutenberg dataset consists of public domain work by authors whereas Reuters dataset consists of financial news articles. Therefore, most of the words of Reuters dataset do not exist in the Gutenberg vocabulary resulting in high number of $< UNK >$ tokens which in turn results in use of backoff probability, and therefore increase in perplexity. The second highest perplexity (for all three - Train, Test, and Dev) belongs to model trained on brown dataset and tested against Reuters. The reason for this is same as above.

TABLE I
Unigram Model

Train Dataset			
	brown	**reuters**	**gutenberg**
brown	1513.8	6780.82	1758.06
reuters	3806.39	1471.21	4882.8
gutenberg	2616.57	12420.1	982.572

TABLE II
Unigram Model

Dev Dataset			
	brown	**reuters**	**gutenberg**
brown	1589.39	6675.63	1739.41
reuters	3808.87	1479.09	4833.88
gutenberg	2604.28	12256.3	991.5

TABLE III
Unigram Model

Test Dataset			
	brown	**reuters**	**gutenberg**
brown	1604.2	6736.6	1762.01
reuters	3865.16	1500.69	4887.47
gutenberg	2626.05	12392.5	1005.79

We can also observe that model trained on Brown dataset performs relatively well on Gutenberg dataset. Brown dataset consists of standard american english texts and Gutenberg consists of public domain work of authors who mostly used common english language. Therefore, there was more overlap between the vocabulary of the two corpus resulting in relatively lower perplexity.

Another observation is that the model trained on reuters performed decently well on other two datasets. This can be because of the presence of common english language terms in the financial news.

III. Implement a Context-Aware Language Model

An n-gram model is a type of probabilistic language model for predicting the next item in such a sequence in the form of a (n - 1) - order Markov model. In this assignment, I have used tri-gram model which is a special case of n-gram with n=3. It takes into account the previous two words (in same order) for prediction of the future word. Formally, probability of a sentence is calculated using the below formula:

$$P(w_1, w_2, \ldots, w_n) = P(w_1)P(w_2|w_1)P(w_3|w_1, w_2)\ldots P(w_n|w_{n-1}, w_{n-2})$$

$$= \prod_{1}^{n} P(w_i|w_{i-1}, w_{i-2})$$

I also introduced the *'START'* and *'END_OF_SENTENCE'* tokens to capture the beginning and end of the sentence. Therefore, the above formula was modified to:

$$P(w_1, w_2, \ldots, w_n) = P(w_1| < START >, < START >)$$
$$* P(w_2| < START >, w_1)P(w_3|w_1, w_2)$$
$$* \ldots$$
$$* P(w_n|w_{n-1}, w_{n-2})$$
$$* P(w_n|w_{n-1}, < EOS >)$$

Along with it, I added tokens to handle special cases when sentence as no words or one word by adding the following conditions:

- <START> <START> <END OF SENTENCE>
- <START> <END OF SENTENCE>

Smoothing is the process of flattening a probability distribution implied by a language model so that all reasonable word sequences can occur with some probability. This often involves broadening the distribution by redistributing weight from high probability regions to zero probability regions. In this assignment, I have used Linear Interpolation smoothing technique. Linear interpolation of trigrams relies on all the three estimates - trigrams, bigrams, and unigrams. The smoothened trigram frequency is defined as:

$$q(w_i|w_{i-1}, w_{i-2}) = \lambda_1 * q_{ML}(w_i|w_{i-1}, w_{i-2})$$
$$+\lambda_2 * q_{ML}(w_i|w_{i-1})$$
$$+\lambda_3 * q_{ML}(w_i)$$

where $\lambda_1 + \lambda_2 + \lambda_3 = 1$ and $\lambda_1 \geq 0, \lambda_2 \geq 0, \lambda_3 \geq 0$

The maximum likelihood estimates of trigram, bigram, and unigram are defined as:

$$q_{ML}(w_i|w_{i-1},w_{i-2}) = c(w_{i-2},w_{i-1},w_i)/c(w_{i-2},w_{i-1})$$

$$q_{ML}(w_i|w_{i-1}) = c(w_{i-1},w_i)/c(w_{i-1})$$

$$q_{ML}(w_i) = c(w_i)/c()$$

where $c(w)$ is the number of times word w is seen in the training corpus, and $c()$ is the total number of words seen in the training corpus.

The three n-gram models - trigram, bigram, and unigram, have different strengths and weaknesses. The trigram model makes use of the context by taking into account the previous two words, however, it suffers from the problem of sparsity. Because of the limited size of the training corpus, words tend to be missing in the test data. The unigram model on the other hand rarely suffers from sparsity and is mainly well-defined but fails to capture the context of the sentence. The bigram model falls between the two extremes. Thus, using linear interpolation, we combine all the three which helps in solving the sparsity problem along with capturing the context.

When a trigram of words (w_1,w_2,w_3) does not exist in the vocabulary of the trained model, the count $c(w_1,w_2,w_3)$ becomes zero and the model relies on bigram and unigram $(\lambda_1 * 0 + \lambda_2 * q_{ML}(w_2,w_3|w_2) + \lambda_3 * q_{ML}(w3))$. Similarly, when both trigram $((w_1,w_2,w_3))$ and bigram $((w_2,w_3))$ cannot be found in the vocabulary the model relies on unigram $(\lambda_1 * 0 + \lambda_2 * 0 + \lambda_3 * q_{ML}(w3))$. To handle the out of vocabulary words, i.e., when w_3 does not belong to unigram, I used back-off probability which is equal to 0.000001. The back-off probability is kept small because if the word is missing from the train corpus, it is most likely to be rare with respect to the corpus.

A. *Hyperparameters*

The hyperparameters of my model (trigram combined with linear interpolation) are the values of λ_1, λ_2, and λ_3 as defined above. To estimate the value of lambdas, I tested the performance of the model for different values of λ_1, λ_2, and λ_3 on the Dev dataset. As shown in *Table IV*, the lowest value of perplexity is achieved when $\lambda_1 \approx \lambda_2 \approx \lambda_3 \approx 0.33$

TABLE IV
Tuning Hyperparmeters

Trigram Perplexity on Dev Dataset			
	brown	**reuters**	**gutenberg**
$(\lambda_1 = 0.34, \lambda_2 = 0.33, \lambda_3 = 0.33)$	742.74	171.01	309.99
$(\lambda_1 = 0.5, \lambda_2 = 0.25, \lambda_3 = 0.25)$	865.422	177.159	336.087
$(\lambda_1 = 0.5, \lambda_2 = 0.3, \lambda_3 = 0.2)$	890.36	176.255	366.878
$(\lambda_1 = 0.6, \lambda_2 = 0.3, \lambda_3 = 0.1)$	1129.08	194.026	385.043

B. *Comparison with Unigram Model*

As shown in Tables V-VII, the trigram model beats the unigram model for all the datasets. There is a 10x decrease in the perplexity for train dataset and 2-3x decrease for test and dev datasets.

TABLE V
Perplexity comparison on all three datasets

Train Dataset		
	unigram	**trigram**
brown	1513.8	13.75
reuters	1471.21	14.48
gutenberg	982.572	20.77

TABLE VI
Perplexity comparison on all three datasets

Dev Dataset		
	unigram	**trigram**
brown	1589.39	742.74
reuters	1479.09	171.01
gutenberg	991.5	309.99

TABLE VII
Perplexity comparison on all three datasets

Test Dataset		
	unigram	**trigram**
brown	1604.2	756.14
reuters	1500.69	176.28
gutenberg	1005.79	313.32

C. *Sampled Sentences*

The sample sentences are obtained using trigram model and linear interpolation with $\lambda_1 = 0.34, \lambda_2 = 0.33, \lambda_3 = 0.33$ and initial word - *the*:

1. Brown
   - Sample 1: mr pastern was with dean bees
   - Sample 2: but after give the for first at once
2. Reuters
   - Sample 1: feb involved heating oil in marketing meeting to exchange their shares on
   - Sample 2: plans for down end were rice bid rate raise of iowa centre west
3. Gutenberg
   - Sample 1: happiness produced temporary alteration in eyes
   - Sample 2: know marianne he did asher at your though humbleness of mind the chariot that our fathers of entered goeth water that have alas lord

of the brook of kidron and it came be obeyed was before your sign

Trigram Model: The samples from 'brown' dataset represent colloquial english sentences. The samples from 'reuters' contain some information related to marketing, shares, and bid. The samples from 'gutenberg' contain some quotes and metaphors which are common in english literature.

Below sample sentences are obtained using unigram model and initial word - *the*:

1. Brown
   - Sample 1: Middle major battle of found populous only you for not as the
   - Sample 2: it time added with art or An would of world of for bunks the becomes to only record the measure you but by
2. Reuters
   - Sample 1: annual Council the came participation and industry
   - Sample 2: kilos Ouko is line this
3. Gutenberg
   - Sample 1: daughter cunning afterward She wait take grown be 13 spread The be they come Lo little
   - Sample 2: as more three And have and brasen good as had 19 house shalt for 22 the towards three less part will want said of after fear these to and could the behaviour thing fulness

Unigram Model: The sentences obtained from unigram model do not follow any particular pattern and it is difficult to comment about the source or type of dataset based on the samples.

D. *Out-of-Domain Text Analysis [Empirical]*

Comparing the results of unigram model (Table I - III) and trigram model (Table VIII - X), we can observe that the perplexity for trigram model is less than that of unigram models for most of the out-of-domain texts. The only case where the perplexity of unigram is less than that of the trigram model is when the model is trained on 'gutenberg' dataset and tested on 'reuters' for all the three - train, dev, and test dataset. The difference in perplexities for this case is approximately 200-300.

TABLE VIII
Trigram + Linear Interpolation Model

	Train Dataset		
	brown	**reuters**	**gutenberg**
brown	13.75	4674.44	1254.06
reuters	2855.1	14.48	4662.47
gutenberg	1914.99	12648.4	20.77

TABLE IX
Trigram + Linear Interpolation Model

	Dev Dataset		
	brown	**reuters**	**gutenberg**
brown	742.74	4586.57	1237.47
reuters	3808.87	171.01	4609.74
gutenberg	1910.71	12426.7	309.99

TABLE X
Trigram + Linear Interpolation Model

	Test Dataset		
	brown	**reuters**	**gutenberg**
brown	756.14	4633.5	1247.56
reuters	2923.91	176.28	4642.62
gutenberg	1912.47	12577.2	313.321

E. *Out-of-Domain Text Analysis [Qualitative]*

The trigram model with linear interpolation generalizes better over most of the datasets and their combinations. The main reason behind this is the use of all the three models - trigram, bigram, and unigram as part of the smoothening. The linear interpolation smoothening is based on weighted average of all the three models and hence offers the best of the three.

As shown above, the only case where the trigram model fails to beat the unigram model is when the model is trained on 'gutenberg' dataset and tested on 'reuters'. The reason behind this is the huge mismatch between the type of dataset. Reuters consists of financial sentences where as Gutenberg consists of english literature. The perplexity became worse for trigram model compared to unigram model because it is harder to find a sequence of three words - (w_{i-2}, w_{i-1}, w_i) which belong to financial sentences in the public domain english literature. The perplexity did not rise exponentially but only by few hundreds because the smoothening technique ultimately falls back to the weighted unigram model when the trigram and bigram are not found.

The perplexities for both trigram and unigram model were very similar for model trained on 'brown' dataset and tested on the 'reuters' dataset. The reason behind this can be same as above. However, the trigram model was slightly better probably because the 'brown' dataset consists of american english texts and it is easier and more likely to find some financial terms in it compared to 'gutenberg' which consists of public work by authors.

IV. References

1. https://en.wikipedia.org/wiki/Perplexity
2. https://towardsdatascience.com/perplexity-in-language-models-87a196019a94
3. https://en.wikipedia.org/wiki/N-gram
4. https://isip.piconepress.com/courses/msstate/ece_8463/lectures/current/lecture_33/lecture_33.pdf
5. http://www.cs.columbia.edu/ mcollins/courses/nlp2011/notes/lm.pdf