Language Modeling

Zhiyu Wang

April 2019

1 Implementation of Language Model

This language model is basically a trigram language model. For each word(including 'END-OF-SENTENCE') in the training set, it will be combined with a first-previous-word and a second-previous-word. For any word that does not have a first-previous-word or second-previous-word, its missing previous word will be replaced by an '$*$' symbol. During the training process, it will store the counts of [second-previous-word + ' ' + word] as well as the counts of [first-previous-word + ' ' + second-previous-word + ' ' + word] into two dictionaries. The conditional log probability is calculated as following: p($w_i|w_{i-1}, w_{i-2}$) = log(count(w_i, w_{i-1}, w_{i-2})/count(w_{i-1}, w_{i-2})). To ensure the language model can properly estimate the probability of rare words and unknown words, laplace smoothing and 'UNK' strategy are used. As a result, when calculate conditional log probability, p($w_i|w_{i-1}, w_{i-2}$) is now equal to log(count(w_i, w_{i-1}, w_{i-2} + 1)/(count(w_{i-1}, w_{i-2} + $|V|$))), where $|V|$ is the size of distinct words(including 'END-OF-SENTENCE') in the training set. In addition, for any word that appear less than α times in the training set, it will be replaced by 'UNK'. As a result, when calculating probability of an unknown word or a word that has been replaced by 'UNK', the model will use count('UNK', w_{i-1}, w_{i-2}).In addition, when the previous word is not in V, it will be replaced by 'UNK'. The size of V, $|V|$ is also changed to size of common words + 1(which is 'UNK'). Since the perplexity keeps going down when a increases, α(the minimum count for common words) cannot be simply tuned according to perplexity. To tune the hyperparemeter a, I have tried numbers from 1 to 5 to see their perplexities on dev set as well as the ratio of words been marked as 'UNK'. Figure 1 shows that perplexities has a linear relationship with α, whereas the slope is flattening after $\alpha = 2$. Figure 2 shows that the ratio of words been marked as 'UNK' is increasing when α increases. By looking at these two graphs, I choose α to be 3 since the ratio is already in the range of [10%,20%], and the perplexities are relatively low.

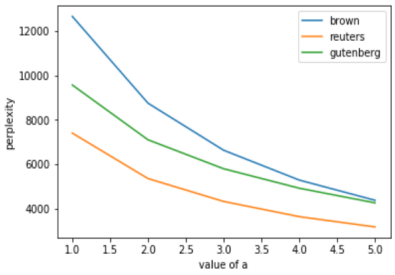

Figure 1: perplexities - α

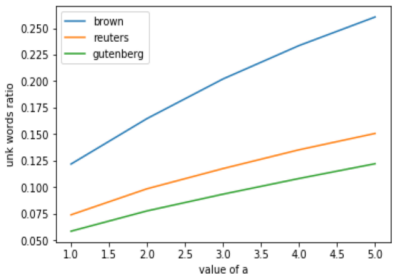

Figure 2: ratio of UNK words - α

2 Analysis on In-Domain Text

2.1 Empirical Evaluation

The perplexity of my model for brown test set is 6630, for reuters test set is 4323, and for gutenberg test set is 5805. This trigram model with smoothing and UNNK has been compared with a simple unigram model and a simple trigram model. The result is stored in the following table. Based on perplexity, the unigram looks better. The reason behind this could be the sentences vary a lot between training sets and test sets. Thus, the patterns appear in training set,(w_i, w_{i-1}, w_{i-2}), do not appear a lot in test set. However, with smoothing and UNK, the perplexity gains significantly improvement compared to basic trigram model, especially for brown dictionary. This might because the brown dictionary contains present American English, which contains many unknown words or unknown patterns in the test set, and the trigram model with smoothing and UNK can handle these cases, whereas the basic trigram model cannot.

Model/Test Set	Brown	reuters	gutenberg
Unigram	1604	1500	1005
Trigram	66169	4336	11897
Trigram with smoothing	6630	4323	5805

When the hyperparemeter α changes, the perplexity changes as well. The result is stored as a graph in Figure 1. It is linearly decreasing when α increases. However, the slope is also flattening. This means as α increasing, it will have fewer impact on the perplexity. In addition, a large α value will cause the dictionary be dominated by UNK. Thus it is important to not only look at perplexity when deciding α.

2.2 Qualitative Analysis

To analysis the quality for different models, I picked 'the' as prefix, and use the generator class with temp = 0.5 to generate different sentences. For unigram model, an example for brown set is " the and to the a", for reuters set

is "the mln a mln the mln", and for gutenberg set is "the the for the her". Apparently these sentences that unigram generated are just random combination of high frequency words. They are not sentences at all. For my model, an example for brown set is "the replacement Abel doses courtyard high Mountain negotiation Revenue periphery weak model expense they Falls Thailand surely corridor poorly devastating 1936 myriad fire comparison joined remainder Caravan clients doubling illiterate disorder", for reuters set is "the subsidized Robins regional Guard Payments range corporation tide underwriter terminated SEEN TENDERED studies PURCHASED Delors subordinate ST desired pfennigs 1997 900 Nov Panamanian DIVIDENDS Jan Nabisco ARC Croo METALS 866 interpretation Xiamen RB RANGES HONDURAS examination", and for gutenberg is "the nineteen ebony panting triangular brutish reminding comeliness Among port aft charade Harim amazed moose sandwiches wildest distinct advance Enscombe addressing Bell Honors weare puzzle". The sentences are much longer, but there are some sequences in the sentences that can make sense now, such as "the nineteen ebony" and "terminated SEEN TENDERED studies". Even though there are still some random combinations of words, it is clearly better than unigram. One problem of my model is the sample sentence is very long, and I cannot set the temp in generator to be very low because UNK would dominate the sentence if I do so.

3 Analysis on Out-of-Domain Text

The perplexities for out-of-domain text for unigram model, trigram model, and trigram with laplace smoothing and UNK are stored in table1,2,and 3. Compared to unigram model, unigram has much smaller In-Domain perplexities, but trigram model with smoothing and UNK gains similar perplexities for Out-of-Domain perplexities. Even more, when unigram model is trained on reuters, it has very high perplexities on the other two dictionaries. This is not obesrved in my model, where model trained on one dictionary has very close perplexities when performed on other two dictionaries. My model also behaves much better than basic trigram model, which has very high perplexities for Out-of-Domain texts.

Using different training sets also leads to different performance. Using reuters dictionary as the training set has the best perplexities when performed on Out-of-Domain text. The other two dictionaries, brown and gutenberg, have similar performance when used as training set. The reason might be that reuters dictionary has a large amount of sentences that can properly represent normal patterns. In addition, it has financial specific sentences where the other two dictionaries does not have. As a result, the other two models trained by either brown or gutenberg cannot predict correctly when perform on reuters, whereas the model trained by reuters can perform properly on all three dictionaries.

Table 1: Unigram Model

Trained Set/Test Set	brown	reuters	gutenberg
brown	1604	6736	1762
reuters	3865	1500	4887
gutenberg	2626	12392	1005

Table 2: Trigam model

Trained Set/Test Set	brown	reuters	gutenberg
brown	66169	299654	158693
reuters	238772	4336	428207
gutenberg	125790	44806	11897

Table 3: Trigam model with smoothing and UNK

Trained Set/Test Set	brown	reuters	gutenberg
brown	6637	4932	6990
reuters	5727	4332	5943
gutenberg	5662	4598	5327

4 Adaptation

The adaptation strategy used in the model is assigning weights to the two-words counts and three-words counts. Basically, after training model on corpus A, the model will train again on a small portion of corpus B. The small portion I use is 1/5 of the corpus B's training set. The training process is very similar to the normal process, which is counting each two-words and three-words occurrence. However, it will not assign any word in corpus B to 'UNK', and any word that has been assigned to 'UNK' based on corpus A will now be replaced by its original word if it occurs in corpus B. In addition, a hyperparemeter, weight W, is used when training on corpus B. For instance, when a combination (w_i, w_{i-1}, w_{i-2}) occurs in corpus B t times, it will be W*t eventually. The W has been set to 20 after several experiments. The result is stored in table 4. When the weight w is small, sometimes the result is worse than the initial model which only trained by corpus A. As w increases, the overall performance gets better than the original model. When W equals 20, the perplexity can be very close to using the whole corpus B as training set. Some perplexities can be even lower than training on corpus B's full training set.

Table 4: Adaptation model

Trained Set/Test Set	brown	reuters	gutenberg
brown	N/A	5021	6301
reuters	6673	N/A	5794
gutenberg	7238	5192	N/A