Statistical NLP 2017 - Assignment 1

Language Models: N-Grams

Daniel Rivera Ruiz

2017-09-11

This assignment is intended to be a first approximation to the analysis of language models through the use of N-Grams. The code provided for the assignment was used to develop a series of experiments in order to evaluate the performance of different models and parameters. In section one the basic unigram model is presented, which will be used as the comparison baseline for more sophisticated models. Sections two and three describe respectively the bigram and trigram models that were used during the development of the assignment. Finally, the results and conclusions of the experiments are discussed in section four.

1 Unigram Model

The code provided for the baseline unigram uses sentences from the Wall Street Journal (WSJ) extracted from the Penn treebank to develop the language model. About 1M words are loaded, which are divided into three sets: 80% training data, 10% validation data and 10% testing data. For the unigram model the validation set is not used, since there are no parameters to tune.

The unigram is the most basic N-gram language model that can be developed: it simply assigns a probability to each word in the training set according to its frequency or count. Let w be a word in the training set, $f(w)$ its frequency and N the size of the set. The probability assigned to w by the unigram model is shown in equation 1:

$$P(w) = \frac{f(w)}{N} \tag{1}$$

To evaluate the model, three indicators are provided: PP_{wsj}, PP_{hub} and WER. PP_{wsj} stands for the perplexity of the WSJ sentences of the validation/test set. PP_{hub} refers to the perplexity calculated on a set of speech recognition problems loaded from the HUB data set. Since this set comes from a source that the model never saw during training, the values for PP_{hub} are expected to be worse (higher) than those for PP_{wsj}. Finally, the WER indicator refers to the *Word Error Ratio* on the HUB recognition task: candidate transcriptions and their acoustic scores are provided, the model combines these acoustic scores with the scores it generates and the best scoring candidates are compared against the correct answers to calculate the WER.

Table 1 shows the values obtained for these indicators with the unigram model. These values will be used in following sections as reference for the bigram and trigram models, in order to evaluate the improvement in their performance.

Table 1: Unigram Model Results.

Model	PP_{wsj}	PP_{hub}	WER
Unigram	1565.2904	1519.1226	0.0914

2 Bigram Model

The bigram language model, and in general all higher order N-grams, build on the notion of assigning probabilities to blocks of consecutive words in the training set. In particular, the bigram model calculates the probability of a

word taking into account only the word right before it (conditional probability). Equation 2 shows how to calculate the probability that a word w_i occurs, given that the previous word is w_{i-1}:

$$P(w_i|w_{i-1}) = \frac{P(w_{i-1}, w_i)}{P(w_{i-1})}$$
$$P(w_i|w_{i-1}) = \frac{f(w_{i-1}, w_i)}{f(w_{i-1})} \tag{2}$$

The problem with this model is that $f(w_{i-1}, w_i)$ can equal zero for bigrams that were not present in the training set, but appear in the validation or test sets. To avoid assigning zero probability in these situations, the interpolation model shown in equation 3 is used, where an amount of probability mass defined by the parameter λ is transferred from the bigram to the unigram probability. This process is called *smoothing*[1] and is used to improve the predictions performed by the bigram model.

$$P(w_i|w_{i-1}) = \lambda * \frac{f(w_{i-1}, w_i)}{f(w_{i-1})} + (1 - \lambda) * \frac{f(w_i)}{N} \tag{3}$$

Using equation 2 (with the training set from the WSJ) a first aproximation to the bigram model is generated. Afterwards, values for PP_{wsj} (on the WSJ validation set), PP_{hub} and WER are calculated. To achieve this, the model from equation 3 is used with different values for the parameter λ.

As it can be observed in figure 1, the best bigram model is obtained with a value of $\lambda = 0.06$. It's also worth noticing that the perplexity values for the HUB set are not higher than those of the WSJ set, as it was expected. This can be explained by the fact that the WSJ validation set contains words that were never seen during the training process, which is not the case for the HUB set.

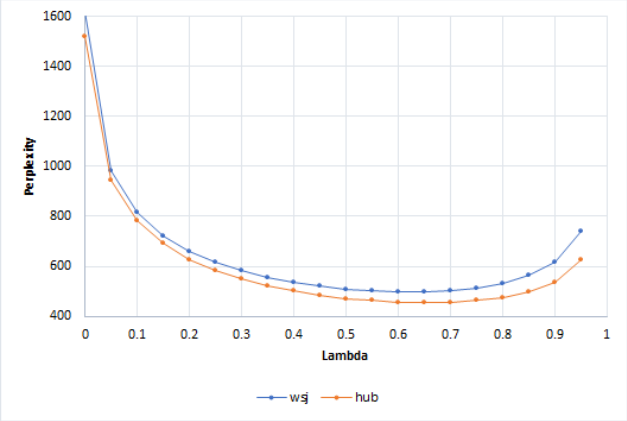

Figure 1: Values for PP_{wsj} and PP_{hub} as λ varies. Bigram Model.

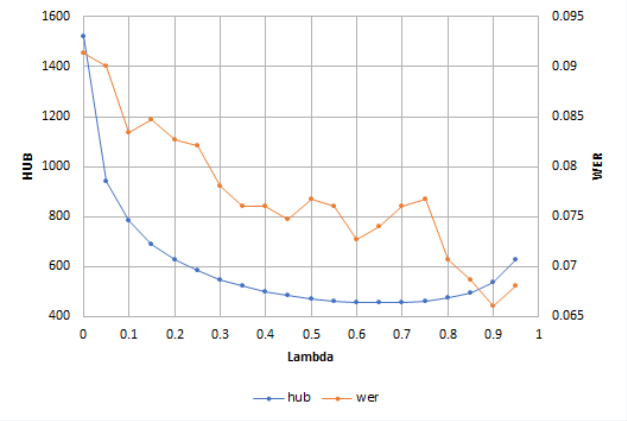

Figure 2: Relationship between PP_{hub} and WER. Bigram Model.

Figure 2 shows the relationship between the two metrics for the HUB set: perplexity and WER. It can be observed that lower values of WER are obtained as λ approaches one. Using such high values of λ would however result in higher perplexities both for the HUB and the WSJ sets, which is why they are not consider to be optimal values for λ.

A linear relationship between PP_{wsj} and PP_{hub} can be observed in Figure 5. A value of $R^2 = 0.993$ for the linear regression model confirms this affirmation and suggests that the values for PP_{wsj} and PP_{hub} are strongly related.

3 Trigram Model

A trigram model is based in the same principle than a bigram model, except that it considers **two** previous words to calculate the probabilities. Using a similar train of thought than the one used for equations 2 and 3, equations

[1] A similar *smoothing* process is also performed for the unigram model, assigning the ***UNKNOWN*** token to all words that were not encountered during the training process.

4 and 5 can be generated for the trigram model:

$$P(w_i|w_{i-1}w_{i-2}) = \frac{f(w_{i-2}, w_{i-1}, w_i)}{f(w_{i-2}, w_{i-1})} \tag{4}$$

$$P(w_i|w_{i-1}w_{i-2}) = \lambda_1 * \frac{f(w_{i-2}, w_{i-1}, w_i)}{f(w_{i-2}, w_{i-1})} + \lambda_2 * \frac{f(w_{i-1}, w_i)}{f(w_{i-1})} + (1 - \lambda_1 - \lambda_2) * \frac{f(w_i)}{N} \tag{5}$$

Since the trigram model depends on two parameters, λ_1 and λ_2, a 3D space would be necessary to plot a graph equivalent to the one shown in figure 1. However, to simplify the visualization of the data, both parameters have been coded into one dimension: tuples of the form (λ_1, λ_2) were generated with values ranging in $[0, 1)$ for both parameters. Values for PP_{wsj}, PP_{hub} and WER were calculated for 100 of these tuples[2] and are plotted in figures 4 and 5.

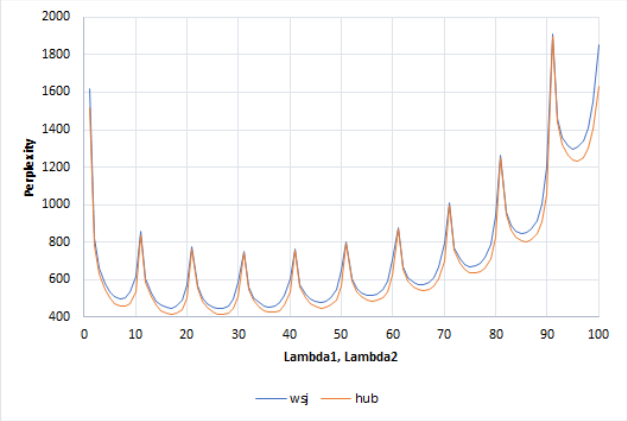

Figure 3: Values for PP_{wsj} and PP_{hub} as λ_1 and λ_2 vary. Trigram Model.

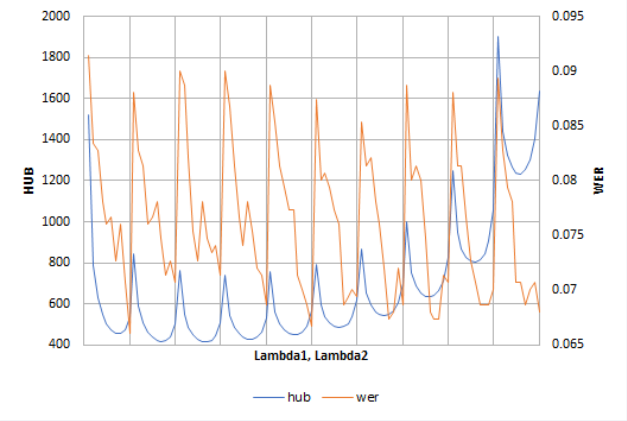

Figure 4: Relationship between PP_{hub} and WER. Trigram Model.

Figure 3 shows that, in general, lower values of the perplexity can be achieved for small values of λ_1. However, as λ_1 increases, better WER scores can be achieved, as figure 4 suggests. Since there is also λ_2 to be considered, it is important to analyze thoroughly all the values generated for the tigram model and select the one that optimizes both perplexities and WER.

Figure 6, similarly to figure 5 for the bigram model, shows the strong linear relationship between PP_{wsj} and PP_{hub} for the tigram model, which is reinforced by the high value of $R^2 = 0.9912$.

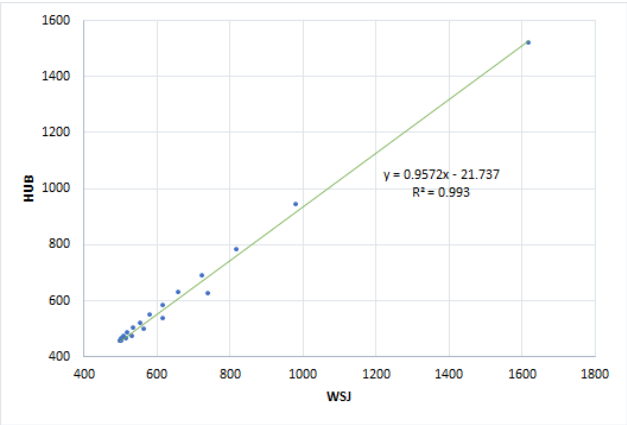

Figure 5: Linear Regression for PP_{wsj} and PP_{hub}. Bigram Model.

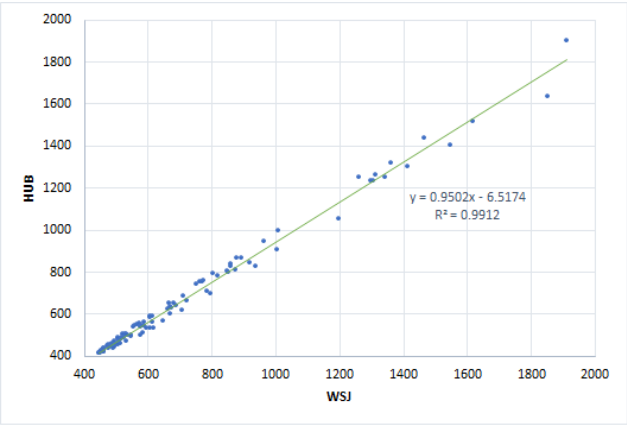

Figure 6: Linear Regression for PP_{wsj} and PP_{hub}. Trigram Model.

[2]The tuples are sorted along the x axis in ascending order for λ_1 first and for λ_2 second.

4 Results and Conclusions

After using the validation set throughout sections 2 and 3 to tune the models parameters, the values that generated the best results were selected and used to generate results for the test set: table 2 shows these results. In the end, three different set of parameters (λ_1, λ_2) were selected for the tigram model, in order to show how, as the model becomes more complex, there might not be a single correct answer, but rather a set of answers that compromise the optimization of some metrics to improve others.

Table 2: Experiment Results.

Model	Parameters λ	PP_{wsj}	PP_{hub}	WER
Unigram	—-	1565.2904	1519.1226	0.0914
Bigram	$\lambda = 0.6$	456.4984	457.2892	0.0727
Trigram 1	$\lambda_1 = 0.1$, $\lambda_2 = 0.54$	407.0187	414.5864	0.0747
Trigram 2	$\lambda_1 = 0.2$, $\lambda_2 = 0.48$	404.0672	413.9854	0.0747
Trigram 3	$\lambda_1 = 0.6$, $\lambda_2 = 0.24$	525.2199	546.4861	0.0674

It is worth noticing that unlike the results obtained for the validation set, the PP_{wsj} values are actually smaller than the PP_{hub} ones. This can be explained by the different amount of unknown words present in the WSJ validation and test sets: the former having 2077 and the latter only 1818. With a smaller number of unknown words, our models perform better in the test set than they did in the validation set.

In general, however, the results confirm what was expected from the experiment: more complex models, when tuned appropriately, can result in better performance of the overall system. This can be observed in improvements of about 70% for the perplexities and of about 20% for the WER values in higher order models compared to the original unigram model. Nonetheless, as the complexity of the model increases, so does its running time, and therefore it is important to take both aspects into consideration when designing language models for real life applications.

References

[1] A. Parikh, "Statistical NLP Fall 2017," September 2017. Available: "http://www.cs.cmu.edu/~apparikh/courses/stat-nlp-fall2017/".

List of Figures

List of Tables