1.0 Forward and Backward Bigram Models

A forward bigram is a specific form of N-gram model which uses one word of prior context. A backward bigram is similar but uses one word of post context. In either case, N-gram models are used to compute the last word from the previous N-1 words (forward) or predict the previous word (backward) from the N-1 words that follow. Dr. Mooney provided the forward bigram java project to be used in creating the backward bigram model.

The backward bigram model can easily be constructed by flipping each input sentence to the train and test functions. By doing so, we can use the same algorithms developed by Dr. Mooney for his forward model, but in the flipped case, the conditioning is occurring on the second token rather than the first token.

The overall algorithm of a bigram is to record the counts for each unigram occurance of a token (word) and each bigram occurance of a pair of tokens. Conditional log probabilities are calculated for the bigram and unigram occurrences. These probabilities are weighted then summed together. A $\frac{1}{10}$ weight is given the the unigram values and a $\frac{9}{10}$ weight is given to the bigram values.

In order to combat out of vocabulary or OOV words during testing, it is important to include a token for unknown words in the training model. In our case, this token is <UNK> which stands for unknown. The <UNK> token allows us to smooth our models. In training we replace the first instance of a token with <UNK>. This allows us to have an open vocabulary system and provides a better model.

Table 1 below reports the nonterminating perplexity results from the forward and backward bigram models. A nonterminating perplexity (or word perplexity) does not predict a sentence boundary (beginning or end.) This metric was chosen to compare the forward and backward model because in English, it's typically easier to predict the beginning of a sentence (backward model) rather than the end of a sentence (forward model.)

Atis	**Forward Bigram**	**Backward Bigram**	**Percent Difference**
Train	11.239	12.111	7.688%
Test	27.039	31.431	15.023%
Brown	**Forward Bigram**	**Backward Bigram**	**Percent Difference**
Train	76.859	75.061	2.367%
Test	334.214	323.235	3.340%

WSJ	Forward Bigram	Backward Bigram	Percent Difference
Train	66.865	65.058	2.739%
Test	288.691	277.805	3.843%

Table 1. Log Scaled Nonterminating Perplexity Using 80% of Corpus for Training

The percent difference was calculated using this formula:

$$Percent\ Diff = \frac{|Perp_{Forward} - Perp_{Backward}|}{AVG(Perp_{Forward,} Perp_{Backward})} \times 100$$

We can determine from the data above that the forward and backward bigram models perform very closely to each other, especially in larger data sets. In larger data sets, the backward bigram barely outperforms the forward bigram with a 2-4% percent difference. The only instance where the forward model performed better than the backward model (with a 7-16% difference between the nonterminating perplexities) was the Atis corpus. It's possible that this occurs because the Atis corprus is constructed from verbal conversations rather than written documents. It could also be caused by the predictability of greeting structures and he types of formulaic requests used to book airlines.

It makes sense that the nonterminating perplexities for the backward and forward model in English are very similar because the same amount of context is used in both instances. Only one word is being conditioned in both cases.

2.0 Bidirectional Models

To construct the bidirectional model, no new code needed to be written. I simply combined the forward and backward models. The only change needed was to recalculate the nonterminating log perplexity. The overall log perplexities were calculated by averaging the perplexities from the forward and backward bigram models (so that both models were equally weighted with a weight of 0.5), but the unigram vs bigram weights (0.1 and 0.9 respectively) from the original word perplexity calculation from Bigram.java (forward model) and BigramModelBackwards. java (backwards model) were kept the same.

Table 2 below reports the nonterminating perplexity results from the forward, backward, and bidirectional bigram models. However, it is important to note that the bidirectional model is technically conditioning on both the previous and past tokens, and is arguably a special trigram as a result.

Atis	Forward Bigram	Backward Bigram	Averaged Bigram Models	Bidirectional Model	Percent Difference between Bidirectional Model and Average Bigram
Train	11.239	12.111	11.675	6.980	50.335%
Test	27.039	31.431	29.235	18.985	42.513%
Brown	**Forward Bigram**	**Backward Bigram**	**Averaged Bigram Models**	**Bidirectional Model**	**Percent Difference between Bidirectional Model and Average Bigram**
Train	76.859	75.061	75.960	21.557	111.58%
Test	334.214	323.235	328.720	100.495	106.346%
WSJ	**Forward Bigram**	**Backward Bigram**	**Averaged Bigram Models**	**Bidirectional Model**	**Percent Difference between Bidirectional Model and Average Bigram**
Train	66.865	65.058	65.962	20.316	105.810%
Test	288.691	277.805	283.25	91.838	102.061%

Table 2. Log Scaled Nonterminating Perplexity Using 80% of Corpus for Training

We can determine from the data above that the bidirectional model vastly outperforms using any one bigram model. The greatest gains are seen in larger and more complex corpora (Brown and WSJ). This is expected as the limitations of the simpler models can be encountered as data sets increase and diversify. Additionally, the bidirectional model is using two tokens to provide context and make predictions (the previous and following words,) so it makes sense that it would perform better when given more data to make predictions.