CS 388 Natural Language Processing
Homework 1: N-Gram Language Model

Victoria Anugrah Lestari (val565)

February 20, 2017

1 Introduction

The purpose of this assignment is to compare the performance of forward bigram, backward bigram, and bidirectional bigram in language modeling. The Java code for the forward bigram model is provided by Professor Ray Mooney. We are supposed to develop the backward bigram model and the bidirectional bigram model based on the existing code. The bigram model is linearly interpolated with the unigram model, using fixed weights for the combination (0.9 for bigram and 0.1 for unigram).

2 Methods

I made both `BackwardBigramModel` and `BidirectionalBigramModel` subclasses of `BigramModel` because they share many of the same methods. The methods that are modified from those of `BigramModel` are overridden.

2.1 Backward Bigram Model

I used the code for bigram model and changed some parts so that it served as a program for backward bigram model. Since we were instructed to think of the simplest way to change the forward model into the backward model, I thought of two simplest ways to reuse the code for bigram model: (1) reversing the sentence (i.e. the list of tokens) or (2) starting the iteration from the last element of the list to the first. In both ways, I switched the sentence-start (<S>) into the sentence-end (</S>) and vice versa.

Since the `BigramModel` code handles the unknown words by labeling the first appearance of a word in a sentence as "<UNK>", the `BackwardBigramModel` labels the last appearance of a word in a sentence as "<UNK>" as well.

For some unknown reasons, the first method generated extremely large word perplexity for the training data (842.87 for the atis corpus) and extremely large perplexity for the testing data (853.92 for the `atis` corpus). Therefore, I used the second method (starting the iteration from the last to first) for my backward bigram code.

2.2 Bidirectional Bigram Model

For the bidirectional model, I used the `sentenceTokenProbs` method from the forward model and the backward model. This method returns the probability of a sentence in the language model, given the unigram distribution and bigram distribution. Then I wrote the modified method `sentenceLogProb` for the bidirectional model, which computes the log of the average probability of each token in the sentence. This method returns the sum of the log as the log probability of the sentence in the model.

The `test` method in the bidirectional model only computes the word perplexity. Because the forward model and backward model are opposite in direction, it does not make sense to include the sentence-start and sentence-end prediction.

3 Results

The backward bigram model generated slightly better results similar to the forward model, except for the `atis` corpus. Table 1 displays the perplexity of the forward model, as shown in the trace file from the homework instruction. Table 2 shows the perplexity and word perplexity of the backward bigram model. It can be said that the forward model and the backward model have similar performance in predicting words in a language model.

Table 1: Perplexity for forward bigram model

	atis	Brown	WSJ
Perplexity (training)	9.043	93.519	74.268
Word perplexity (training)	10.592	113.360	88.890
Perplexity (testing)	19.341	231.302	219.715
Word perplexity (testing)	24.053	310.667	275.119

I think the backward model performs slightly better than the forward model because given the words in a sentence except one (the first or the last), it is easier to guess the first word in a sentence than the last word in a sentence. For example, given the sentence "__ eat spaghetti", it is easy to guess that the first word is "I". However, given the sentence "I eat __", it is not that obvious what the last word should be. It could be "spaghetti", "noodle", "meatball", etc.

Table 2: Perplexity for backward bigram model

	atis	Brown	WSJ
Perplexity (training)	9.013	93.509	74.267
Word perplexity (training)	11.636	110.783	86.660
Perplexity (testing)	19.364	231.206	219.520
Word perplexity (testing)	27.161	299.686	266.351

The word perplexity of the bidirectional bigram model is significantly lower than that of the forward model or the backward model for all three corpora. In other words, the bidirectional model has the best performance. The reason is that given a word in a sentence, this model knows the word previous to it and the word next after it. Table 3 shows the word perplexity of the bidirectional model.

Table 3: Perplexity for bidirectional bigram model

	atis	Brown	WSJ
Word perplexity (training)	7.235	61.469	46.514
Word perplexity (testing)	12.700	167.487	126.113