CS388: N-gram Language Models Project Report

Zeyuan Hu
Computer Science Department
University of Texas at Austin
Austin, Texas
iamzeyuanhu@utexas.edu

Abstract

In this project, we build a backward bigram model and a bidirectional language model. We compare these two models with (forward) Bigram model and find that the bidirectional language model achieves the best result in terms of perplexity in all three corpora: ATIS-3 (Dahl et al., 1994), Penn Treebank (Marcus et al., 1994), and Brown (Francis and Kucera, 1979).

1 Introduction

N-Gram language model has many applications in NLP. The idea of N-gram model is to estimate probability of each word given previous $N-1$ as prior context. Specifically, for bigram model, we calculate $P(w_k|w_{k-1})$ to estimate the probability of appearance for a given word w_k. Once we have the probability estimate for each word, we can estimate the probability of the appearance for a given sentence. The standard (forward) N-gram language model models the generation of text from left to right. However, we can have better prediction of tokens (i.e., words) based on the context to their right. That is the motivation for the backward bigram model. In addition, for a given token, we might be at best estimate its probability both from its left context and its right context, which leads us to the bidirectional language model.

2 Implementation Details

We build our backward bigram model and bidirectional bigram model based on the source code provided by Ray Mooney (Mooney, 2018). Since the only difference between backward bigram model and bigram model is that backward bigram model models the generation of a sentence from right to left. Thus, we make a separate class file `Ngrams` to hold the share methods between two models. Then, we extend `Ngrams` class to implement both `BigramModel` and `BackwardBigramModel`. We also implement `BidirectionalBigramModel` in the same way as the previous two models. In `BidirectionalBigramModel`, we linearly interplote the probability estimate of a given token from `BigramModel` and `BackwardBigramModel` with equally weights (i.e., $P(w_k) = \lambda_1 P(w_k|w_{k-1}) + \lambda_2 P(w_k|w_{k+1})$ `where` $\lambda_1 + \lambda_2 = 1$).

Beyond the implementation approach highlighted above for both `BackwardBigramModel` and `BidirectionalBigramModel`, we need to think about dealing with $\langle S\rangle$ and $\langle /S\rangle$. For a given sentence $\langle S\rangle ABC\langle /S\rangle$, whether we reverse $\langle S\rangle$ and $\langle /S\rangle$ in the `BackwardBigramModel` does not matter: we can have $\langle S\rangle CBA\langle /S\rangle$ or $\langle /S\rangle CBA\langle S\rangle$ as long as we stick with one convention consistently in both training and testing phase(i.e., perplexity calculation). One thing to note is that when we use $\langle S\rangle CBA\langle /S\rangle$, we modify the semantics of $\langle S\rangle$ to indicate the end of sentence and $\langle /S\rangle$ as the beginning of the sentence. In the implementation, we empirically verify this point by exposing additional input variables `start_marker` and `end_marker` to both the training and evaluation functions. In addition, we use linear interpolation with a unigram model to perform smoothing and we replace the first ouccrence of each token with $\langle UNK\rangle$ in both training and testing sets to handle out-of-vocabulary (OOV) words.

3 Experiments and Analysis

We run all three models on all three corpus with parameter setup shown in Table 1. Table 2 shows

the word perplexity of three models on both training and testing data from three datasets. Table 3 shows the perplexity of bigram and backward bigram models. Word perplexity is different from perplexity in the sense that we exclude the probability estimate of $\langle S \rangle$ of each sentence in bigram model and the probability estimate of $\langle /S \rangle$ in the backward bigram model. Doing so allows us to compare the models in a resaonable way (i.e., generating $\langle S \rangle$ is a task different from generating $\langle /S \rangle$).

As one can see from Table 2, there is not any significant difference in terms of mode performance between bigram model and backward bigram model. However, the word perplexity does drop from 275.12 to 266.35 in Penn Treebank test set and 319.67 to 299.69 in Brown corpus test set. This slight performance increment is partially due to the fact that right context of a token has slightly more information than its left context. However, the similar performance of bigram model and backward bigram model indicates that the left context word and right context word of a token contains roughly same information for the token prediction, which suggests that we should treat equally when we combine them together.

In bidirectional bigram model, the word perplexity drops significantly across all three models on the test set with 46% drecrease in ATIS-3, 54% in Penn Treebank, and 48% in Brown corpus when compared with bigram model word perplexity. The experiment result hints that we can indeed make better predication of words given the context words surrounding them. This finding implies that the bidirectional model incorporate more information about corpus than using the bigram or backward bigram model alone and we can improve model by including information about corpus as much as we can. To verify if we should treat both bigram and backward bigram model equally in bidirectional bigram model, we vary λ_1 and plot the word perplexity of the bidirectional bigram model on the test set of Penn Treebank and as shown by Figure 1, the observation holds.

4 Conclusion and Future Work

In this project, we implement backward bigram model and bidirectional bigram model and we show that using the surround words as context can give us the highest chance to predict a given token correctly. In the future, we can see whether this

Table 1: Parameter setup

Description	Values
training data split	0.9
testing data split	0.1
Weights of unigram in smoothing	0.1
Weights of bigram in smoothing	0.9
Weights of bigram in birdirectional λ_1	0.5
Weights of backward bigram in bidirectional λ_2	0.5

Table 2: Word Perplexity Comparison

Model	data type	ATIS-3	Penn Treebank	Brown
Bigram	train	10.59	88.89	113.36
	test	24.05	275.12	319.67
Backward	train	11.64	86.66	110.78
	test	27.16	266.35	299.69
Bidirectional	train	7.24	46.51	61.47
	test	**12.70**	**126.11**	**167.49**

Table 3: Perplexity Comparison

Model	data type	ATIS-3	Penn Treebank	Brown
Bigram	train	9.04	74.26	93.52
	test	**19.34**	219.72	231.30
Backward	train	9.01	74.27	93.51
	test	19.36	**219.52**	**231.21**

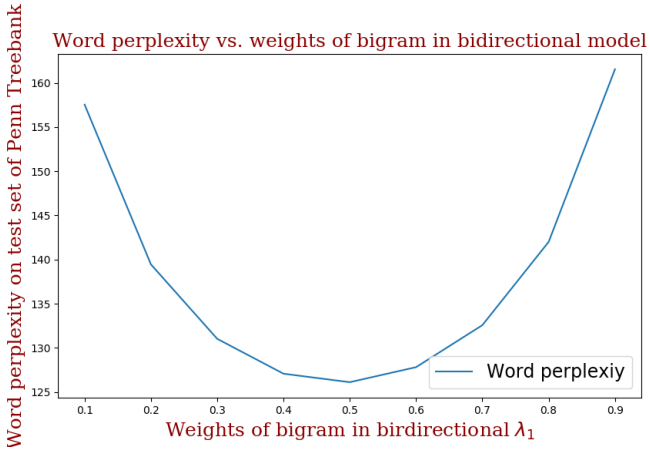

Figure 1: Word perplexity vs. λ_1

observation still holds under bidirectional-LSTM setting.

References

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-smith, David Pallett, Er Rudnicky, and Elizabeth Shriberg. 1994. Expanding the scope of the atis task: the atis-3 corpus. In *in Proc. ARPA Human Language Technology Workshop '92, Plainsboro, NJ*. Morgan Kaufmann, pages 43–48.

W. N. Francis and H. Kucera. 1979. Brown corpus manual. Technical report, Department of Linguistics, Brown University, Providence, Rhode Island, US. http://icame.uib.no/brown/bcm.html.

Mitchell Marcus, Grace Kim, Mary Ann Marcinkiewicz, Robert MacIntyre, Ann Bies, Mark Ferguson, Karen Katz, and Britta Schasberger. 1994. The penn treebank: Annotating predicate argument structure. In *Proceedings of the Workshop on Human Language Technology*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT '94, pages 114–119. https://doi.org/10.3115/1075812.1075835.

Ray Mooney. 2018. CS388 Natural Language Processing Homework 1: N-gram Language Models.