CS 388 Natural Language Processing

Homework 1: N-gram Language Models

Jianyu Huang (jh57266)

1. Comparative Results

Table 1 "Perplexity" for the training and test data given the first two models

		atis	wsj	brown
Bigram	Training	9.043	74.268	93.5193
	Test	19.341	219.715	231.302
BackwardBigram	Training	9.013	74.268	93.5091
	Test	19.364	219.520	231.206

Table 2 "Word Perplexity" for the training and test data given all three models

		atis	wsj	brown
Bigram	Training	10.592	88.890	113.36
	Test	24.054	275.118	310.667
BackwardBigram	Training	11.636	86.660	110.783
	Test	27.161	266.352	299.686
BidirectionalBigram	Training	7.235	46.5144	61.469
	Test	12.700	126.113	167.487

2. Discussions

We use the first 90% of the sentences for training and the other 10% for test.

The "Word Perplexity" of the backward bigram model (for both training and test data) is quite similar to the normal bigram model. The backward bigram model is equivalent to estimate the probability of each word based on the next word, while the normal bigram model uses the previous word. So either backward bigram model or normal bigram model uses only one of prior context or later context. Since the probability of predicting the next word by the current word and deriving the current word by the next word is almost the same, the "word perplexity" of the backward bigram model is similar to the normal bigram model. However, there are some tiny differences. For small data set "atis", the "Word Perplexity" of the normal bigram model (for both training and test data) is a little better than that of the backward bigram model, while for larger data set "wsj" and "brown", the "Word Perplexity" of the backward bigram (for both training and test data) model is slightly better than that of the normal bigram model. My guess for the reason for the better performance of the backward bigram in larger data set is that the end of the sentence is more predictable than the start of the sentence.

The "Word Perplexity" of the bidirectional model (for both training and test data) is better than both the backward model and the normal model. On both training and test data, the "word perplexity" of the

bidirectional model is almost half of that of the backward model or the normal model. N-Gram Models is based on the Markov assumption that the future behavior of a dynamic system only depends on its recent history. So in N-Gram Model we give the estimated probability of each word based on (N-1) words of prior context. However, the estimated probability of each word should also be based on the later context. While either backward bigram model or the normal bigram model uses only one of the prior context or later context, the bidirectional model will use both of the prior and the later contexts. So the bidirectional bigram model estimates the probability of each word based on both the previous word and the next word. It will be much more accurate to estimate the probability of the word in the middle in some phrases by bidirectional model.

The "Word Perplexity" of the training data is better than the test data for all three models. This is because we get the models with the training data.

The "Perplexity" is better than "Word Perplexity" for both training and test data given the normal bigram model and backward bigram model. That results from that the "Word Perplexity" measure excludes the prediction for the end-of-sentence </S> for the normal bigram model or the start-of-sentence <S> for the backward bigram model.

3. Trace Files

bigram-trace.txt

```
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BigramModel /projects/nlp/penn-treebank3/tagged/pos/atis/ 0.1
# Train Sentences = 519 (# words = 3922)
# Test Sentences = 58 (# words = 431)
Training...
Perplexity = 9.043192013019798
Word Perplexity = 10.5919539986291
Testing...
Perplexity = 19.341388622483773
Word Perplexity = 24.053999598153656
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BigramModel /projects/nlp/penn-treebank3/tagged/pos/wsj/ 0.1
# Train Sentences = 43820 (# words = 995626)
# Test Sentences = 4869 (# words = 111718)
Training...
Perplexity = 74.26799183241482
Word Perplexity = 88.89008380713945
Testing...
Perplexity = 219.71517777445175
Word Perplexity = 275.1178149885602
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BigramModel /projects/nlp/penn-treebank3/tagged/pos/brown/ 0.1
# Train Sentences = 47207 (# words = 1079440)
# Test Sentences = 5245 (# words = 93530)
Training...
Perplexity = 93.51927720192262
```

```
Word Perplexity = 113.35954408376686
Testing...
Perplexity = 231.30243689356243
Word Perplexity = 310.66735613437913
```

backward-bigram-trace.txt

```
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BackwardBigramModel /projects/nlp/penn-treebank3/tagged/pos/atis/ 0.1
# Train Sentences = 519 (# words = 3922)
# Test Sentences = 58 (# words = 431)
Training...
Perplexity = 9.0129362437796
Word Perplexity = 11.636203016013203
Testing...
Perplexity = 19.364374738655066
Word Perplexity = 27.16138806179997
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BackwardBigramModel /projects/nlp/penn-treebank3/tagged/pos/wsj/ 0.1
# Train Sentences = 43820 (# words = 995626)
# Test Sentences = 4869 (# words = 111718)
Training...
Perplexity = 74.26778753071389
Word Perplexity = 86.6602647021424
Testing...
Perplexity = 219.51980597986474
Word Perplexity = 266.3515745993808
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BackwardBigramModel /projects/nlp/penn-treebank3/tagged/pos/brown/ 0.1
# Train Sentences = 47207 (# words = 1079440)
# Test Sentences = 5245 (# words = 93530)
Training...
Perplexity = 93.50913083897301
Word Perplexity = 110.78289581654053
Testing...
Perplexity = 231.20551767464033
Word Perplexity = 299.68570342320015
```

bidirectional-bigram-trace.txt

```
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BidirectionalBigramModel /projects/nlp/penn-treebank3/tagged/pos/atis/
0.1
# Train Sentences = 519 (# words = 3922)
# Test Sentences = 58 (# words = 431)
Training...
Word Perplexity = 7.235173934082009
Testing...
Word Perplexity = 12.700210156018738
```

```
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BidirectionalBigramModel /projects/nlp/penn-treebank3/tagged/pos/wsj/
0.1
# Train Sentences = 43820 (# words = 995626)
# Test Sentences = 4869 (# words = 111718)
Training...
Word Perplexity = 46.514445090631405
Testing...
Word Perplexity = 126.1131573880389
jianyu@sloth:~/Work/Github/NLP/hw1$ java -cp /u/jianyu/Work/Github/NLP/hw1
nlp.lm.BidirectionalBigramModel /projects/nlp/penn-treebank3/tagged/pos/brown/
0.1
# Train Sentences = 47207 (# words = 1079440)
# Test Sentences = 5245 (# words = 93530)
Training...
Word Perplexity = 61.46886647115352
Testing...
Word Perplexity = 167.48711091425574
```