Homework 1: N-gram Language Models

Yu Feng

2-16-2015

1 Introduction

In this homework, my task is to produce both a "backward" bigram model and a bidirectional model based on the sample code of a normal bigram model.

2 Algorithm

2.1 Backward Bigram Model

The algorithm for the backward bigram model is almost identical to the normal bigram model except for the model direction. So what I need to do is to swap the model direction for both the training and testing data and then invoke the original corresponding functions.

2.2 Bidirectional Bigram Model

Here are the steps to build the bidirectional bigram model:

Step 1: Create an instance of the bidirectional model which has references to the instances of the bigram model and the backward bigram model;

Step 2: Train each model separately by invoking its own training method using the given data;

Step 3: When determining the probability of each word(α_i) using the bidirectional model, linearly interpolate the predicted results of the bigram model and the backward bigram model based on the following equation:

$$\alpha_i = log(\lambda_1 * \beta_i + \lambda_2 * \gamma_i)$$

Here, β_i and γ_i are the probability predicted by the bigram model and the backward bigram model, respectively. λ_1, λ_2 are their corresponding weight. I weight them equally in the experiment.

Step 4: The probability for the entire sentence will be the sum of all its words:

$$\Psi = \sum_{i=1}^{n} \alpha_i$$

where n is the number of words in current sentence.

POS Data	Bigram	Backward	Bidirectional
atis	10.59	11.64	7.24
wsj	89.02	86.76	46.58
brown	110.26	107.65	59.83

Table 1: Comparison of Word Perplexity among three models on the training data from the atis, wsj and brown corpora.

POS Data	Bigram	Backward	Bidirectional
atis	24.05	27.16	12.70
wsj	265.50	257.16	121.25
brown	433.77	423.88	232.01

Table 2: Comparison of Word Perplexity among three models on the testing data from the atis, wsj and brown corpora.

3 Experiments

I implemented the algorithms described in section 2 and tested them on the atis, wsj and brown corpora. Table 1 and Table 2 shows the overall results for word perplexity.

The results show that the backward bigram model outperforms the normal bigram model in the *wsj* and *brown* corpora, but worse in the *atis* corpora; Meanwhile, bidirectional bigram model yields the best word perplexity across all POS data sets.

To reproduce the results, first compile the sources by running the "`compile.sh`" script and execute the main method of each model accordingly. For example, executing the following command will dump the trace for the backward bigram model on the *wsj* corpora:

```
java -cp bin/ nlp.lm.BackwardBigramModel pos/wsj/ 0.1
```

which will display the trace similar to this:

```
# Train Sentences = 43820 (# words = 995024)
# Test Sentences = 4869 (# words = 112320)
Training...
Perplexity = 74.36070352512667
Word Perplexity = 86.75945534817288
Testing...
Perplexity = 212.41719625520673
Word Perplexity = 257.1577316631044
```

Running the "`batch.sh`" script will display the comparison on all POS data sets.

4 Discussion

RQ1:How does the "Word Perplexity" of the backward bigram model (for both training and test data) compare to the normal model?

Intuitively speaking, give a large corpora, it's hard to tell which model is better since the only difference is the model direction. I think their performances are related to the characteristics of the corpora: the backward model may be better in some corpora but worse otherwise. The results of table 1 and table 2 are consistent with this point. But we still need more data and different parameters to verify this point.

RQ2:How does the "Word Perplexity" of the bidirectional model (for both training and test data) compare to both the backward model and the normal model?

Table 1 and Table 2 show that the bidirectional model is strictly better than both the backward model and the normal model in terms of "Word Perplexity". This is also reasonable because the bidirectional algorithm is taking advantage of the knowledge from both sides. When predicting each token, the bidirectional model will have constraints from forward and backward thus the results outperform any of those two.