LANGUAGE MODELING EXPERIMENTS

Program #8

EE 8993: Speech Processing

Audrey Le

Professor: Dr. Joseph Picone

July 31, 1998

1. PROBLEM DESCRIPTION

In this project, we are to perform simple experiments to learn more about standard language modeling techniques. Four sets of experiments will be performed, each exploring some characteristics of language modeling. The corpus to be used in the language modeling experiments is the Wall Street Journal data collected from 1993 to 1994.

In the first set of experiments, we are to generate a histogram of word unigrams, bigrams, and trigrams from our corpus, compute the entropy of each N-gram distribution, and discuss the nature of the distributions. We want to see what kind of data we have. By knowing the characteristics of the data, we will be more informed in choosing the parameters to build our language model.

In the second set of experiments, we want to find the out-of-vocabulary (OOV) rate as a function of the N most frequent words. In this set of experiments we want to determine the optimum dictionary size given the available data. We do not want a dictionary that covers all the words since such dictionary is so large that the search time is impractical.

For the third set of experiments, we select the most frequent 1,000 words and compute the trigrams from this vocabulary. We want to determine how many the trigrams generated from the 1,000 word vocabulary cover our corpus.

For the last set of experiments, we are to partition our corpus into two sets, a training set and a test set. The training set is to contain 80% of the corpus, and the test set is the remaining data in the corpus. We are to build a language model from the training data. The language model is evaluated using the test set. This is repeated for three more partitions of the data. We can use any method to partition the data.

2. INTRODUCTION

The task of a speech recognition system is to produce the word sequence W given a sequence of acoustic observables A [1]. The word sequence that has the maximum posterior probability is the sequence that the recognizer outputs. This is summarized in Equation (1).

$$\hat{W} = \arg max_{W}\{Pr(W|A)\} \tag{1}$$

where $\hat{W}$ is the predicted word sequence, and $Pr(W|A)$ is the probability of the word sequence given the acoustics.

Using Bayes' law, we can rewrite Equation (1) as:

$$Pr(W|A) = \frac{Pr(W)Pr(A|W)}{P(A)} \tag{2}$$

where $Pr(W|A)$ is the probability of the word sequence given the acoustics, $Pr(W)$ is the prior probability of producing the word sequence, $Pr(A|W)$ is the probability of the acoustics given the word sequence, and $Pr(A)$ is the probability of the acoustics.

By substituting Equation (2) into Equation (1), we have Equation (1) as:

$$\hat{W} = \arg max_W \frac{Pr(W)Pr(A|W)}{P(A)} \tag{3}$$

$Pr(W)$ is the probability of the word sequence, $Pr(A|W)$ is the probability of the acoustics given the word sequence, and $Pr(A)$ is the probability of the acoustics. Since $Pr(A)$ is constant, it can be omitted. Thus Equation (3) becomes:

$$\hat{W} = \arg max_W (Pr(W)Pr(A|W)) \tag{4}$$

The task of the language model is to provide the $Pr(W)$ in the Equation (4) above. We can obtain $Pr(W)$ by using the definition of conditional probability.

$$P(W) = Pr(w_1 \ldots w_M) = \prod_{m=1}^{M} Pr(w_m | w_{m-1} \ldots w_1) \tag{5}$$

According to Equation (5), the probability of the word sequence W is the product of the probability of observing a word w_m given its immediate n predecessor words $w_{m-n} \ldots w_{m-1}$. The resulting model is a Markov chain and is referred to as $n+1$ -gram model. For $n = 0$, we obtain the unigram model, $n = 1$ we have bigram model, $n = 2$ we have the trigram model and so on.

The $n+1$ -gram model, commonly known as the N-gram, is estimated from the text corpus during the training phase. However, most of the possible events are never seen in the training because there are so many of possible N-grams. Therefore, in order to allow for events not seen in training, the probability distributions obtained in these N-gram approaches are smoothed with more general distributions. Smoothing is a method that is used to counteract the effect of statistical variability as they turn up in particular in small training sets. In addition to smoothing, other techniques such as discounting, backing-off, and interpolation are also used to deal with this sparse data problem [2].

In this assignment, we will learn various aspects of language modeling by experimenting with N-gram models, in particular, unigram, bigram, and trigram models.

3. EXPERIMENTS AND RESULTS

The first experiment attempts to determine the characteristics of the corpus that will be used in

training and testing our language model. The purpose of this experiment is to inform us of the type of data we have available. Once we have an idea of the kind of data available, we will be informed in our choice of parameters that go into training a language model.

We generated the unigrams, bigrams, and trigrams from the corpus using the CMU-Cambridge Statistical Language Modeling Toolkit. Each list of N-grams was sorted in decreasing order, and the results are shown in Figure 1. Because the number of bins is so large, only the top 1,000 bins are displayed. Due to limitation in our graphing tool, if all the bins were displayed, all we will see is a straight line for each distribution.

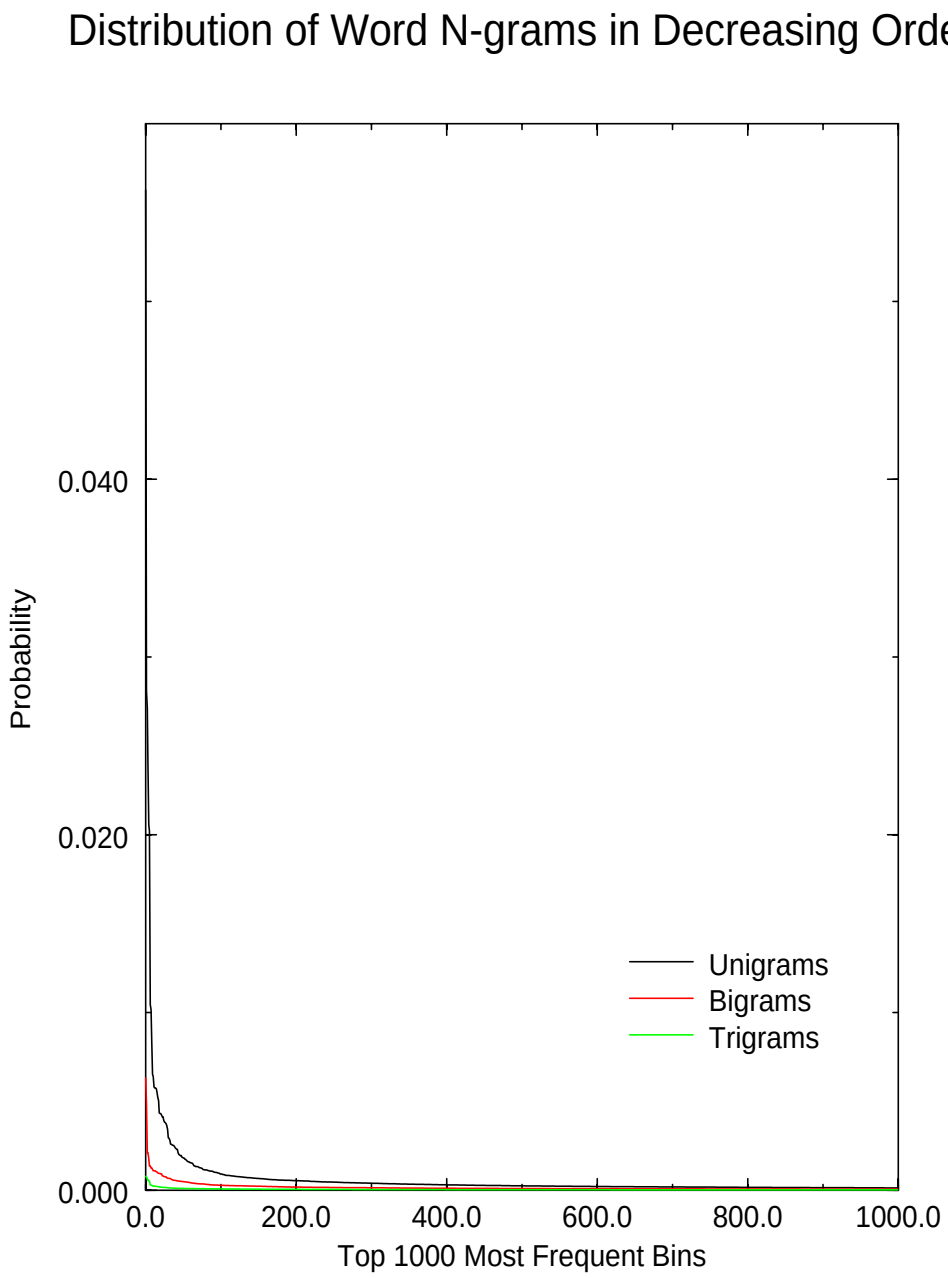

Figure 1: Distribution of Word N-grams for the Top 1000 Bins.

We note from the results that trigrams have lower probability than bigrams or unigrams. This makes sense since for N unique words there are N^3 possible unique trigrams, N^2 bigrams, and N unigrams. Thus, the probability of a trigram is likely be lower than that of a bigram or unigram, and the probability of a bigram is likely to be lower than that of a unigram. This observation is further confirmed by the entropy calculated from each distribution. From the table below, the trigram distribution has the highest entropy of the three distributions followed by that of the bigram distribution and theunigram distribution. Even with a large corpus, in our case approximately 25 million word corpus, it contains only a fraction of the possible trigrams. Thus, unseen or infrequent events are likely to be estimate incorrectly. Some type of smoothing must be used to provide better estimates.

N-gram	Entropy
Unigram	7.650499
Bigram	12.789862
Trigram	15.501179

Table 1: Entropy of word N-gram distribution.

In the second experiment, we wanted to determine the OOV as a function of N most frequent words. We took the top N words in the corpus and determined how much they covered the corpus using Equation (6).

$$OOV = \frac{l}{M} \times 100 \tag{6}$$

where l is the total number of words in the corpus that are not found in the dictionary and M is the total number of words in the corpus. The results are given in Figure 2.

From the results in Figure 2, we can see that the top 1,000 words cover approximately 70% of the vocabulary. Although 1K vocabulary is very small, it covers the majority of our corpus. If we were to include more words in our dictionary then the OOV will likely to decrease further. However, in LVCSR systems, it is impossible to include all the words due to space limitation. In addition, the search time for such a large vocabulary is impractical even with the aide of the best algorithm and faster processor. Thus, a balance must be strike between space and speed. Often the optimum size of the vocabulary is determined empirically based on the data that are available.

Since the top 1,000 words cover the majority of the vocabulary of the corpus, how many trigrams generated from this vocabulary cover the corpus is the subject of the third experiment. We generated 1 billion possible trigrams from this vocabulary and determined their coverage on our corpus. The results are given in Table 2.

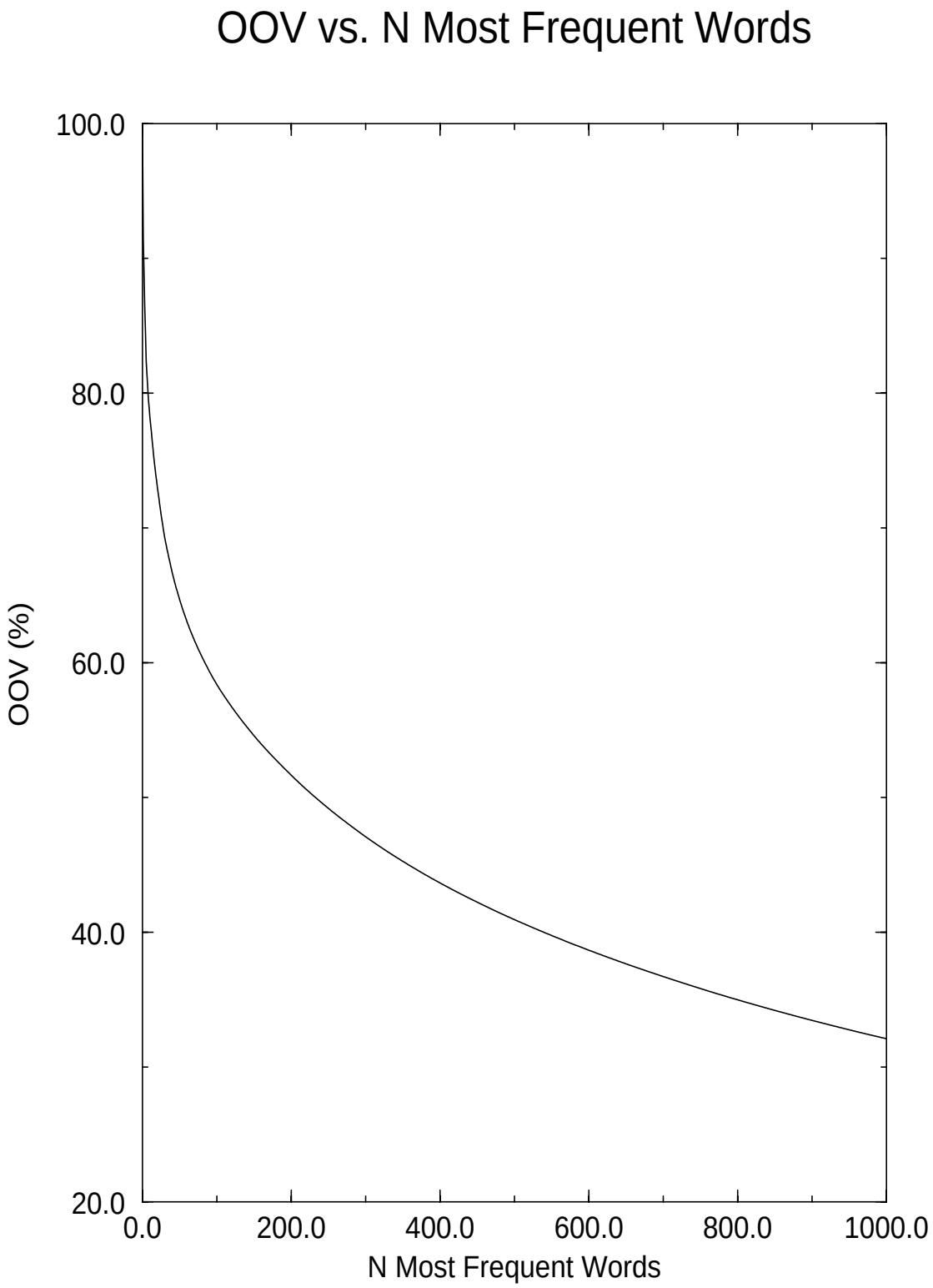

Figure 2: OOV rate as a function of N most frequent words.

Vocabulary	No. of Unique Words	No. of Trigrams	Percent Coverage
1000 most frequent words in corpus	1,000	1e+9	86.51%
corpus	301,362	14,209,687	100%

Table 2: Trigram coverage of various vocabulary sizes.

In the next experiment, we partitioned the text corpus into two sets—a training set and a test set—and use these sets to build and evaluate the language model.

We used the uniform random number generator to pick the training and test sets. The random number generator generated an index which points to an item in the corpus. For our case, an item is a paragraph. The algorithm works as follows. If the paragraph has not been picked, it is put in the test set and is marked off as "been picked". If that paragraph has been picked, it is discarded and not put in the test set. Another index is generated, and the process continues until the test set has approximately N percent of the corpus where N is a variable indicated by the user. For our case, N is 20% of the corpus. The remaining data in the corpus is assigned to the training set. For each partition, the random number generator is seeded so that each partition will produce different training and test sets. We used this algorithm to generate three additional training and test sets.

For each partition of the corpus obtained using the algorithm above, a trigram language model was built using the training set, and the model was evaluated using the test set. This was repeated for three other partitions. The results are given in Table 3.

Partition	OOV	Perplexity	Entropy	No. of 3-grams Hit	No. of 2-grams Hit	No. of 1-gram Hit
1	4.74	179.87	7.49	59.97	31.58	8.64
2	4.73	179.94	7.49	59.95	31.59	8.46
3	4.73	180.58	7.50	59.90	31.62	8.48
4	4.73	180.03	7.49	59.95	31.58	8.47

Table 3: Statistics of various language models trained and evaluated on different train and test sets.

The results of the four sets are highly correlated. One possibility is that the random selection procedure is not truly random. However, on examining the content of the sets, we have found no patterns in these four sets. We can only suggest that the data from which we partitioned these sets is highly correlated. Thus, we suggest a language model that combines these four language models using an equal set of weights since these four language models have fairly similar results.

4. CONCLUSIONS

We have presented here the results of our experiments with language modeling and have sug-

gested an interpolate language model based on our experiments. We have found that even with different partitionings of the corpus, the performance did not improve. It is possible that our data is highly correlated or that our data is too small to make reliable estimation. We concluded that additional data are needed to determine the exact cause.

5. REFERENCES

[1] J. Picone, *ECE 8993 Speech Processing Lecture Notes*, Mississippi State Univ., Mississippi State, MS, 1998.

[2] P. Clarkson and R. Rosenfeld, "Statistical Language Modeling Using the CMU-Cambridge Toolkit," *Proc. of Eurospeech,* 1997.