SCALABLE BACKOFF LANGUAGE MODELS

Kristie Seymore *Ronald Rosenfeld*

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

ABSTRACT

When a trigram backoff language model is created from a large body of text, trigrams and bigrams that occur few times in the training text are often excluded from the model in order to decrease the model size. Generally, the elimination of n-grams with very low counts is believed to not significantly affect model performance. This project investigates the degradation of a trigram backoff model's perplexity and word error rates as bigram and trigram cutoffs are increased. The advantage of reduction in model size is compared to the increase in word error rate and perplexity scores.

More importantly, this project also investigates alternative ways of excluding bigrams and trigrams from a backoff language model, using criteria other than the number of times an n-gram occurs in the training text. Specifically, a difference method has been investigated where the difference in the logs of the original and backed off trigram and bigram probabilities is used as a basis for n-gram exclusion from the model. We show that excluding trigrams and bigrams based on a weighted version of this difference method results in better perplexity and word error rate performance than excluding trigrams and bigrams based on counts alone.

1. INTRODUCTION

Current collections of text for statistical language model training are making the sparse training data problem less serious for certain domains, such as ARPA's Wall Street Journal corpus, which is part of the 305 million word North American Business News collection. The more training text that is used for language model creation, the more unique word sequences are encountered that must be stored in the model. Thus, as training text size increases, language model size necessarily increases, which can lead to models that are too unwieldy and memory-demanding to be of practical use. This overabundance of training data will allow us, or more correctly force us, to be selective in choosing the training data that we use to create our models. We explore two methods of training text pruning that allow for compact and efficient creation of trigram backoff language models. The effects of the original amount of training data on a scaled-down model is also investigated.

2. THE BACKOFF LANGUAGE MODEL

The backoff language model was developed by Katz [2] to address the problems associated with sparse training data. Small counts result in unreliable estimates. The backoff model handles this type of sampling error by discounting the probability of low count events and distributing the freed probability mass among unseen events.

As the amount of training text used to create the backoff model increases, the number of unique trigrams and bigrams increases. The language model will necessarily takes up more memory in order to store the additional information from the training text. At some point, the model's memory requirements will exceed any practical system capacity. Therefore, we can either limit the amount of training data we use to develop the model, or take from a large amount of training text that portion which leads to the most reliable word predictions.

3. PRUNING TECHNIQUES

Word sequences that occur the fewest number of times in a training text can lead to unreliable predictions. This idea has led to the popular cutoff method of training text reduction, where only information about the most frequently occurring bigrams and trigrams is included in the language model. This method will be explored in depth in Section 3.1. However, we also need to consider the word sequences for which the model would not make a good prediction if they were eliminated from the model. This idea has led to the development of the weighted difference method of training text reduction, which will be introduced in Section 3.2.

3.1. The Cutoff Method

The cutoff method of training text pruning excludes from the language model those bigrams and trigrams that occur infrequently. The motivation for this method lies in the argument that there is not much difference between a trigram or bigram occurring once in a text of millions of words and it not occuring at all. Just by excluding those n-grams with a count of one from a model, a significant savings in memory can be achieved. In a typical training text, roughly 80% of unique trigram sequences occur only once. This idea can be extended further to bigrams and trigrams that occur any number of times. We can designate a trigram cutoff and a bigram cutoff, and all bigrams and trigrams that occur the same number of times or less than their cutoff are excluded from the backoff language model.

What kind of memory savings can we expect from excluding bigrams and trigrams in this manner? In Carnegie Mellon University's Sphinx II speech recognizer, each trigram takes up 4 bytes of memory and each bigram takes 8 bytes (because it contains a backoff weight and a pointer to the dependent trigrams.) The memory required for unigram probabilities and constants can be considered a fixed overhead, and is not included in our memory calculations. Using a 58,000 word dictionary and 45 million words of Wall Street Journal training data (1992 – 1994), the memory requirements of models created with different cutoffs can be computed. Several sample model sizes are shown in Table 1, with cutoffs indicated by (bigram cutoff – trigram cutoff). A cutoff of k means that n-grams occuring k or fewer times are discarded. For this data, 78.5% of the trigrams and 61% of the bigrams occur only once, so we see that significant memory savings can be obtained by cutting out the bigrams and trigrams that appear infrequently.

Model Cutoffs	# Bigrams	# Trigrams	Memory (MB)
(0–0)	4,627,551	16,838,937	104
(0–1)	4,627,551	3,581,187	51
(1–1)	1,787,935	3,581,187	29
(0–10)	4,627,551	367,928	38
(10–10)	347,647	367,928	4

Table 1: Model Cutoffs and Resulting Model Size

In order to investigate the effects of raising bigram and trigram cutoffs, several models were created using the Carnegie Mellon Statistical Language Modeling Toolkit [4]. The word error rate (WER) and perplexity (PP) were calculated for each model. The perplexities of the scaled down models were computed using the official ARPA 1994 Language Model Development Set, and the word error rate was computed using CMU's Sphinx II system and the ARPA 1994 Hub 1 Acoustic Development Set (7387 words). Several models were created by pruning only trigrams, while others incorporated bigram and trigram pruning. First, the amount of trigrams to be retained in the model was determined, and then the cutoff was set to be the maximum cutoff possible so that all trigrams with a count equal or less than the cutoff plus some number with a count of (cutoff+1) were removed from the model. The trigrams cut out at level (cutoff+1) were the first ones encountered in an alphabetized list. For combined bigram and trigram pruning, the number of bigrams retained in the model was as close as possible to the number of trigrams in the model. The bigram and trigram cutoffs were chosen so that these desired totals could be met, resulting in bigram and trigram cutoffs that were not necessarily the same.

3.2. The Weighted Difference Method

If an n-gram is not present in the model, the model uses a backed off probability estimate in place of the original estimate. If that backed off estimate is very close to the original estimate, then there is not a need to store the original estimate in the first place. This idea has led to the weighted difference method of training text reduction.

The weighted difference factor of an n-gram is defined to be

$$w.d.factor = \\ K * (\log(\text{original prob}) - \log(\text{backedoff prob})) \quad (1)$$

where K is the Good-Turing discounted n-gram count. This factor reflects our desire to keep an n_gram in the language model.

The CMU Statistical Language Modeling Toolkit was modified to create weighted difference language models by pruning n-grams based on their weighted difference factor. Several models were created. The results are plotted with the cutoff method results, and are shown in Figures 1 - 4. In both cases, as the language model size is decreased, the perplexity rises sharply. Trigram pruning does not have much effect on WER, but bigram and trigram pruning causes memory savings and increases in WER to become significant.

As can be seen from these figures, the models created with the weighted difference method have significantly lower perplexity values than for the cutoff models, but the perplexity rises in the same manner in both cases. The word error rates for the weighted difference models are almost always lower than that of the cutoff models, but the significance of the difference is questionable. We can say with confidence that using the weighted difference method is at least as good as the cutoff method, and generally yields improved perplexity and word error rates over the cutoff method.

Table 2 displays more clearly the results depicted in Figure 4 for the weighted difference method, with the relative increase in WER over the original (0–1) model shown.

# Bigrams	# Trigrams	Memory (MB)	WER (increase)
4,627,551	3,581,187	51 MB	(original model)
4,627,551	400,000	39 MB	1% relative
4,627,551	70,000	37 MB	3% relative
934,351	900,000	11 MB	5% relative
416,338	400,000	5 MB	9% relative
108,117	100,000	1.3 MB	20% relative

Table 2: Model Reduction and Resulting WER Increases

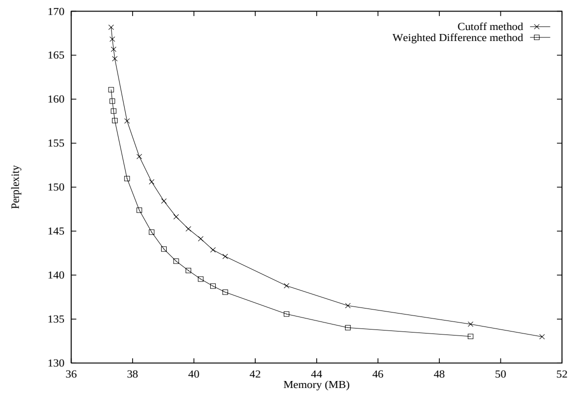

Figure 1: Perplexity vs Scaled Language Model Size, Trigram Pruning Only, 1992 - 1994 Data.

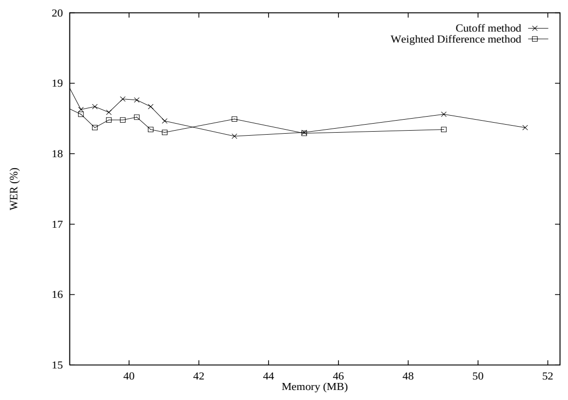

Figure 2: Word Error Rate vs Scaled Language Model Size, Trigram Pruning Only, 1992 - 1994 Data.

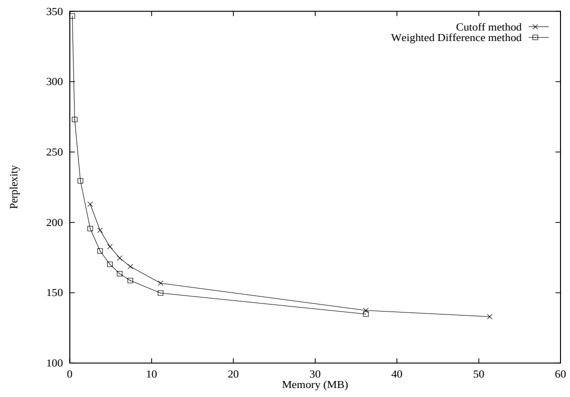

Figure 3: Perplexity vs Scaled Language Model Size, Bigram and Trigram Pruning, 1992 - 1994 Data.

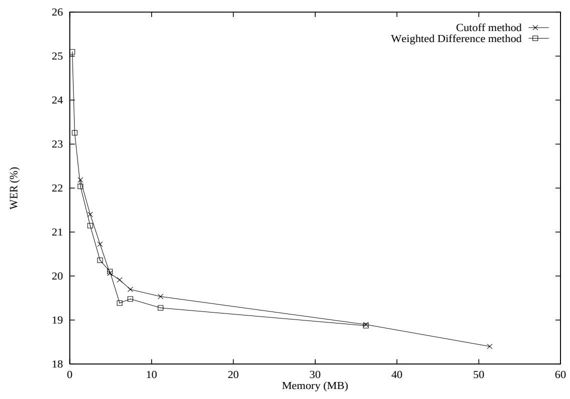

Figure 4: Word Error Rate vs Scaled Language Model Size, Bigram and Trigram Pruning, 1992 - 1994 Data.

Is model size reduction a feasible practice? We see in Table 2 that significant memory reduction can be achieved. Certainly, for particular applications, the increase in WER is worth the savings in memory.

4. EFFECTS OF DIFFERENT AMOUNTS OF TRAINING DATA

In order to verify that using more training data and then pruning it down is a better approach than just starting with a smaller body of training data, three different sized data sets were defined and used to create models of the same size. The first data set consists of 45.3 million words of Wall Street Journal data (1992 - 1994), the same data set whose results are shown above. The second data set is a subset of the first data set, consisting of 28.5 million words of Wall Street Journal data from 1993 - 1994. The third set is yet a smaller set, 6.5 million words of 1994 Wall Street Journal data. Several language models of approximately the same size were computed with the three data sets using both the cutoff and weighted difference methods, pruning as many bigrams and trigrams as necessary in order to reach the desired size. For the third set of data (6.5 million words), the largest memory data point represents a (0-0) model, where no pruning has occurred at all. For all three sets, the weighted difference method generally outperformed the cutoff method in terms of perplexity and word error rate.

Figures 5 and 6 show the weighted difference results for all three data sets. It can clearly be seen that the 6.5 million word models perform significantly worse than the models originally created from 45.3 and 28.5 million words. The difference between the first and second data sets is not as significant, yet the larger data set does do slightly better.

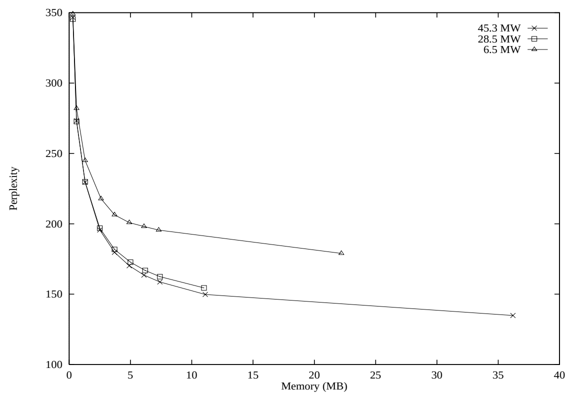

Figure 5: Perplexity vs Scaled Language Model Size for Different Amounts of Training Data (Weighted Difference Pruning.)

There are several factors that need to be considered when analyzing the results of Figures 5 and 6. First of all, the three data sets do not come from the same distribution. There is a time shift present, in that the data that is added to the 6.5 million words to get the 28.5 and 45.3 million words is older data. If a significant change of style or content has occurred over time for that source, the statistics of the older data may be less helpful in modeling probabilities due to bigram and trigram frequencies that do not accurately reflect the current frequency distributions of the language source. In fact, we found a consistent 10% perplexity increase when the 6.5 MW of 1994 data was replaced by a comparable amount of 1992 data. In previous work ([3]), we found a similar effect on the OOV rate.

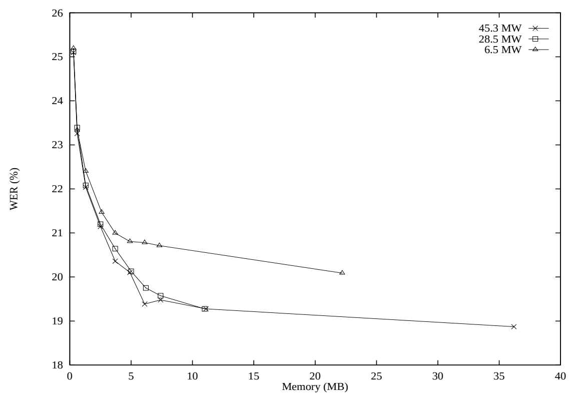

Figure 6: Word Error Rate vs Scaled Language Model Size for Different Amounts of Training Data (Weighted Difference Pruning.)

Second, there seems to be a threshold at about the 1–2 MB model size for which the perplexities and word error rates degrade equally no matter how much data was used initially. At some point, so much information has been pruned from the model that perhaps the models converge to approximately the same set of bigram and trigram sequences, which are those that occur the most frequently. For example, 92% of the bigrams and 87% of the trigrams are the same in the two 1.3 MB models based on 28.5 MW and 45.3 MW. Further intersecting with the 6.5 MW model yields a 73% bigram and a 59% trigram overlap. Using approximately the same set of bigrams and trigrams with approximately the same set of probabilities is likely to lead to similar performance.

5. CONCLUSION

From the results presented in the previous sections, we can conclude that, at least in this domain:

- Training text pruning can be used to build compact and efficient language models that require significantly less memory than language models built from complete training text.
- As model size decreases, the weighted difference method of training text pruning results in a significantly smaller perplexity increase than the cutoff method.
- As model size decreases, the weighted difference method of training text pruning generally results in a slightly smaller word error rate increase than the cutoff method.
- Using more training data, up to at least 25 - 30 million words initially, and then pruning it down is a better approach than just starting with a small amount of training data, as long as the training text does not contain significant style changes and the pruning is not severe (at least 2MB remaining). Beyond 25 million words, the amount of training data does not have a noticeable effect.

Further analysis, detailed results and ideas for future investigation are presented in [5].

6. ACKNOWLEDGMENTS

This material is based upon work supported under a National Science Foundation Graduate Research Fellowship and by the Department of the Navy, Naval Research Laboratory under Grant No. N00014-93-1-2005. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government or the National Science Foundation.

7. REFERENCES

1. F. Jelinek, “Self Organized Language Modeling for Speech Recognition”, in *Readings in Speech Recognition*, Alex Waibel and Kai-Fu Lee (Eds.), Morgan Kaufmann, 1989.
2. S.M. Katz, “Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer”, in *IEEE Transactions on Acoustics, Speech and Signal Processing*, volume ASSP-35, pages 400-401, March 1987.
3. R. Rosenfeld, “Optimizing Lexical and N-gram Coverage Via Judicious Use of Linguistic Data”, Eurospeech 95, pp. 1763–1766. See file://localhost/afs/cs.cmu.edu/user/roni/WWW/vocov-eurospeech95-proc.ps.
4. R. Rosenfeld, “The CMU Statistical Language Modeling Toolkit, and its use in the 1994 ARPA CSR Evaluation”, in *Proc. ARPA Spoken Language Technology Workshop*, Austin, TX, January 1995. See http://www.speech.cs.cmu.edu/speech/SLM_info.html.
5. K. Seymore and R. Rosenfeld, “ Scalable Trigram Backoff Language Models”, *Carnegie Mellon University Tech Report CMU-CS-96-139*, May 1996.