Evaluation of a Language Model using a Clustered Model Backoff

John W. Miller and Fil Alleva

Microsoft Corporation
One Microsoft Way
Redmond WA 98052, USA

ABSTRACT

In this paper, we describe and evaluate a language model using word classes automatically generated from a word clustering algorithm. Class based language models have been shown to be effective for rapid adaptation, training on small datasets, and reduced memory usage. In terms of model perplexity, prior work has shown diminished returns for class based language models constructed using very large training sets. This paper describes a method of using a class model as a backoff to a bigram model which produced significant benefits even when trained from a large text corpus. Tests results on the Whisper continuous speech recognition system show that for a given word error rate, the clustered bigram model uses 2/3 fewer parameters compared to a standard bigram model using unigram backoff.

1. INTRODUCTION

The use of class based language models has previously been reported by many researchers [1,2,3]. Typically the use of class based language models has been confined to applications where insufficient source text is available. This includes those applications requiring dynamic adaptation of the lexicon and language model and to those applications with significant memory constraints.

In this paper a backoff model using clustered bigrams is evaluated. The baseline system for comparison is a standard bigram model using unigram backoff probabilities [4]. The class based system is created by modifying the baseline system by in effect replacing the unigram probabilities with class bigram probabilities. The class bigram probability is the probability of the word given its class times the probability of a word given the class of the prior word. The probabilities are estimated using the empirical frequencies in the training set. A clustering algorithm is used to assign each word to a class so as to minimize the perplexity of the resulting model [3,5]. After a clustering algorithm creates the classes from a training corpus, a series of models of differing sizes and perplexities is then created by omitting counts less than a threshold. This makes it possible to effectively compare clustered versus non-clustered models with identical perplexity. It was found that the clustered backoff model achieves a 50% reduction in the number of required parameters for a given perplexity.

In order to demonstrate the effectiveness of the clustered backoff model we have incorporated it into the large vocabulary Whisper [6] CSR system the configuration of which corresponds to the 19,982 word 1992 Wall Street Journal domain. Our experiments show that the class based backoff model achieves between 4.8% and 15.3% reduction in word error rate for bigram count cutoffs between 1 and 50 respectively. For models with identical error rate, the data shows a 2/3 reduction in the number of parameters in the model.

2. CLASS LABELS FROM CLUSTERING

The clustering algorithm is a search procedure which finds a class label for each word. A set of words with the same class label is called a cluster. During the search, different assignments of words to classes are compared by evaluating a language model built with the different class label assignments.

2.1. The Clustered Bigram Language Model

Define G(w) to be the class assignment for a word. For any given assignment of words to classes, it is simple to collect from a training corpus of text the empirical frequency of each word F(w), of each class F(c), and of the empirical frequency of a word from one class being followed immediately by a word from another $F(c_{i-1},c_i) = F(G(w_{i-1}),G(w_i))$. From these frequencies the probability of a word given the prior word can be estimated as:

$$P''(w_i|w_{i-1}) = P(c_i|c_{i-1}) \cdot P(w_i|c_i)$$

$$P''(w_i|w_{i-1}) = \frac{F(c_{i-1},c_i)}{F(c_{i-1})} \cdot \frac{F(w_i)}{F(c_i)}$$

This estimate is called the clustered bigram model. Using this model, the probability of any sequence of N words of text can be estimated by the product:

$$P(w_1, w_2, \ldots w_n) = \prod_{i=1}^{N} P''(w_i|w_{i-1})$$

The goal of the clustering algorithm is to maximize the estimated probability of a training corpus. Maximizing the product of these probabilities is equivalent to minimizing the geometric mean of the inverse of the probability estimates. This geometric mean is called the perplexity of model P'' on the corpus $w_1, w_2, \ldots w_N$. The perplexity for this model can be written as:

$$PP(w_1, w_2, \ldots w_N) = 2^{(-1/N)\cdot\sum_{i=1}^{N} \mathrm{Log}(P''(w_i|w_{i-1}))}$$

2.2. Clustering to Minimize Perplexity

The algorithm for generating class labels is iterative optimization clustering as outlined in [7]. Each word is initialized to a random cluster (class label). On each iteration each word from a large set of words is moved to the class which produces the model with minimum perplexity. The perplexity modifications are calculated independently, so that each word is evaluated as if all other word classes are held fixed. The algorithm quickly converges so that no single word can be moved to another class in a way which reduces the perplexity of the clustered bigram model.

The number of clusters used for the language model experiments reported in this paper was 256. It was found that slightly better (lower perplexity) models are created by a refinement upon the iterative optimization in which the algorithm is first run with only 32 classes. After the algorithm converges with each word assigned to one of 32 classes, 256 classes are created by randomly assigning these words in each class to one of 8 subclasses. The clustering algorithm is then run with the restriction that words are only moved to one of the 8 subclasses associated with each class. Only after this converges is the clustering algorithm run again with the search extended to placing each word in any of the 256 classes which minimize the perplexity. Examination of the resulting classes shows that this refinement helps to create some classes which are composed of many low probability words. This is in contrast to the direct clustering into 256 classes which tends to produce clusters such that each cluster includes at least one relatively high probability word.

In order to make the clustering algorithm fast it is necessary to limit the number of scans through the large training corpus. This is accomplished by collecting the frequency that each word is followed by each class and the frequency that each word is preceded by each class. From these counts and from the unigram counts it is possible to quickly calculate the change in training set perplexity which occurs when a single word is moved from one class to another.

3. CLUSTERED BACKOFF MODELS

A standard bigram backoff model is a model which combines the estimated word pair probability: $P(w_i|w_{i-1}) = F(w_i,w_{i-1}) / F(w_{i-1})$, with a unigram probability $P(w) = F(w) / N$. The backoff model uses the bigram probability times a parameter slightly less than one (called the discount weight) unless this estimate is zero in which case it uses the unigram probability:

$$\tilde{P}(w_i|w_{i-1}) = \begin{cases} B(w_{i-1}) \cdot F(w_i) / N & \text{if } k = 0 \\ A(k) \cdot k \ / \ F(w_{i-1}) & \text{else} \end{cases}$$

where we define for notational convenience:

$$k = F(w_{i-1}, w_i)$$

The discount weights A(k) are a function of the bigram count so that the probability estimates based upon very small counts are reduced more than the accurate probability estimates based on large counts. The precise weights are chosen by the lowest perplexity model created from the equation form $A(k) = 1 - \lambda^k$. The normalization parameter $B(w)$ is set to the constant that makes the marginal probability of a word equal to the sum calculated from the conditional probabilities:

$$\sum_{w_{i-1}} \tilde{P}(w_i|w_{i-1}) \cdot F(w_{i-1}) / N = F(w_i) / N$$

The clustered backoff model $\tilde{P}_C$ is the model created by replacing the unigram backoff $F(w_i) / N$ with the clustered language model probability P''.

$$\tilde{P}_C(w_i|w_{i-1}) = \begin{cases} B_C(w_{i-1}) P''(w_i|w_{i-1}) & \text{if } k = 0 \\ A_C(k) \cdot k \ / \ F(w_{i-1}) & \text{else} \end{cases}$$

The discount weights A_c and normalization parameters B_c are chosen in the same way as a standard backoff model.

Any bigram model can be made smaller by setting small bigram counts to zero: if $k < T$, then replace k with zero in the equations. As the threshold T is increased the perplexity of the model increases, but the memory required to store the model is decreased. In Figure 1. the perplexity of a test set (the test set does not includes any text used in the training corpus) is plotted as a function of the number of bigrams for the two different backoff models. The perplexity of the clustered backoff model is lower than the standard unigram backoff model even when half as many bigrams are used in the clustered model. This shows that the small improvements in perplexity translate into large reductions in the amount of memory required for a model with given perplexity.

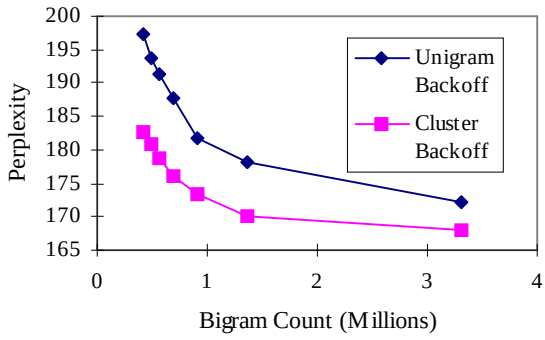

Figure 1: Bigram count vs. perplexity for bigram models using a unigram backoff or a clustered backoff model.

4. SPEECH RECOGNITION WITH THE CLUSTERED BACKOFF MODEL

In the abstract, replacing a unigram backoff with a clustered backoff in a speech recognition system is straight forward. This is the case for a linearly structured search space where each state is associated with exactly one word. However for tree structured search spaces where each state corresponds to a unique pronunciation prefix and only the leaves of the tree correspond to unique word identifiers care must be taken not to introduce search errors or unnecessarily expand the search space.

4.1. Overview of Tree Structured Search Spaces

To achieve efficiency in a single pass decoder the lexicon is represented as a *pronunciation prefix tree* (PPT) [2]. The PPT representation arose from the observation that most of the search effort expended was on the initial phones of the words. The PPT representation reduces that effort by evaluating common phonetic prefixes only once. There are two problems that have arisen when using a PPT, first, if the tree is re-entrant then only the single best linguistic context is considered when the tree is initialized at each hypothesized word boundary and second, the application of the language model is delayed since the identity of the word is only known at the leaves of the tree.

4.2. Re-entrant vs. Non Re-entrant Trees

Using a re-entrant PPT has no adverse implications if the language model to be employed is a unigram since the decision regarding the best word is a local one. (Actually this is true only if no cross word co-articulation models are used, however that is beyond the scope of this paper). Likewise if the system is targeted towards isolated word recognition then word boundaries can reliably be determined and an arbitrary language model can be applied to the resulting lattice. However, for continuous speech recognition systems using higher order language models the linguistic state cannot be determined locally and the word boundaries are uncertain. Several solutions based on creating copies of the PPT for each unique linguistic context solve this problem [8,9,10], however these approaches create redundant sub-tree computations, the number of which correspond to the number of active linguistic contexts. A computation is redundant when a sub-tree instance is dominated by another instance of that sub-tree. A sub-tree instance is dominated when the best outcome in that sub-tree is worse than the worst outcome in another instance of that sub-tree, See minimax search [11].

4.3. Using LM Probabilities in a PPT

The second problem presented by the use of a PPT is that a unique word identity is not determined until a leaf of the tree is reached, as a result it delays the application of the language model probability until the leaf is reached and this generally results in greater search effort. Aubert et.al. proposed [12] that the probabilities be *smeared* over the PPT. While this is adequate for the case of the unigram language model the application of higher order models still must be delayed until a leaf of the PPT is reached. To overcome this difficulty Antoniol et.al.[9] used *successor trees* where higher order language model probabilities are distributed over the appropriate tree. However successor trees do not eliminate the redundant computations revealed by subtree dominance.

To handle the problem of redundant computations the state space of the search is limited to a single copy of the lexical tree where each state is augmented with a heap of linguistic contexts. The size of this heap can then be controlled with an additional heuristic pruning threshold. For more details see [13].

4.4. Clustered Backoff and Tree Structured Search Spaces

The clustered backoff model presents the same problem as 2-gram and higher order language models. That is the word identification is not known until the leaf of the tree is reached. But this can be addressed by creating one PPT tree per class as defined by the clustered backoff model. The drawbacks to this approach are one, the increased memory required by the N trees formed by the N clusters since they will occupy more space than the single unigram tree, and two, the increased search space. Note that for linear search spaces this problem is non-existent due to the sub-optimal search space organization.

Our approach to the problem is to retain the unigram tree and maintain a heap of linguistic contexts at each state corresponding to the active class and word contexts. Once a leaf is reached (and the identification of word w_i, and therefore cluster c_i is known) the probability of the path is updated with $P(c_i \mid c_{i-1})$. The heap mechanism and associated pruning heuristic which previously proved effective for controlling the search space for a unigram backed-off bigram has kept the expansion of the search space to only a 10% increase.

5. EXPERIMENTAL RESULTS

The speech recognition system processes 16kHz pcm data using a mel-scale cepstrum which is then decomposed into several feature streams which are quantized into a stream of weighted codebook indices. A semi-continuous, gender dependent model consisting of 6000 senones is then used to obtain the acoustic likelihoods. The phonetic modeling in the system consists of position and context dependent within word and cross word triphones. composed of three hidden Markov states. The search organization was evaluated on the male portion of the 1992 development test set for the Wall Street Journal corpus. A more complete description of the Whisper speech recognition system can be found in [6]. Figure 2. shows the word error rates for a bigram model using clustered backoff as a function of the number of bigram parameters used in the model. Also given for comparison is the word error rate for the standard bigram model using unigram backoff. As the figure shows, the clustered bigram model is able to achieve the same word error rate with approximately 40% as many parameters as the standard backoff model.

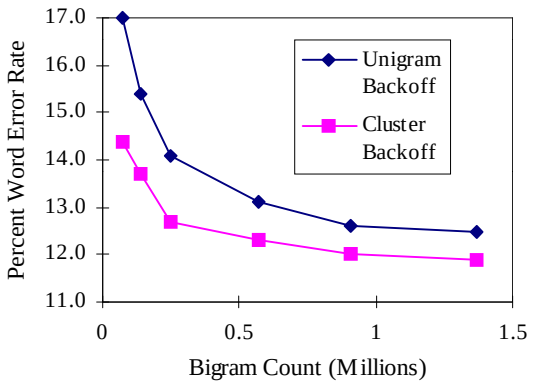

Figure 2: Bigram count vs. word error rate for bigram models using a unigram backoff or a cluster backoff model.

6. SUMMARY CONCLUSIONS

We have presented a class language model based on word clustered bigrams that reduces by at least a factor of 2.7, the number of parameters required by the language model to achieve a given error rate. In addition we have shown how that language model can efficiently be used in the context of a decoder based on pronunciation prefix trees.

Future work will address how word clustered class n-gram models can be adapted as well as augmented with novel words.

7. REFERENCES

1. F. Jelinek. "Self-Organized Language Modeling for Speech Recognition", *Readings in Speech Recognition*, A. Waibel, K.F. Lee, Morgan Kaufmann 1990.

2. H. Ney, R. Haeb-Umbach, B.-H. Tran, and M. Oerder. "Improvements in Beam Search for 10000 Word Continuous Speech Recognition". *Proceedings of the IEEE ICASSP*, March 1992.

3. B. Suhm and A. Waibel. "Towards Better Language Models for Spontaneous Speech". *ICSLP* 1994.

4. S. Katz, "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer". *IEEE Trans. ASSP*. Vol. ASSP-35, p. 400. March 1987.

5. R. Kneser and H. Ney. "Improved Clustering Techniques for Class-Based Statistical Language Models", *EURO-SPEECH 93*, Berlin, Vol. 2, p 973.

6. X. Huang, A. Acero, F. Alleva, M. Hwang, L. Jiang, and M. Mahajan. "From Sphinx-II to Whisper - Making Speech Recognition Usable". *Speech and Speaker Recognition-Advanced Topics*, Kluwer Publisher, 1994.

7. R. Duda, and P. Hart. *Pattern Classification and Scene Analysis*, Wiley, NY, 1973.

8. H. Ney, "Modeling and Search in Continuous Speech Recognition", *ESCA 3rd EuroSpeech*, September 1993.

9. G. Antoniol, F. Brugnara, M. Cettolo, M. Federico, "Language Model Representations for Beam-Search Decoding". *Proceedings of the IEEE ICASSP*, Detroit, MI, 1995.

10. J. Odell, V. Valtchev, P. Woodland, and S. Young. "A One Pass Decoder Design for Large Vocabulary Recognition", *Proceedings of the ARPA HLT Workshop*, Plainsboro, NJ, 1994.

11. N. Nilsson, *Problem Solving Methods in Artificial Intelligence*, McGraw-Hill Book Co., NY, 1971.

12. X. Aubert, C. Dugast, H. Ney, V. Steinbiss. "Large Vocabulary Continuous Speech Recognition of Wall Street Journal Data", *Proceedings of the IEEE ICASSP*, Adelaide, South Australia, 1994.

13. F. Alleva, X. Huang, and M. Hwang. "Improvements on the Pronunciation Prefix Tree Search Organization". *Proceedings of the IEEE ICASSP*, Atlanta, GA, 1996.

8. APPENDIX

To determine the effectiveness of replacing the unigram with a clustered bigram backoff we tested the word error rates for both models for different cutoff thresholds. Table 1 below shows the word error rates for thresholds 1, 2, 4, 12, 25, 50, and *infinity*.

Cut Off	Retained Bigrams	Unigram Backoff %Error Rate	Clustered Backoff %Error Rate	%Error Rate De-crease
infinity	0	28.0	18.8	32.9
50	74,708	17.0	14.4	15.3
25	136,977	15.4	13.7	11.0
12	250,204	14.1	12.7	9.9
4	568,170	13.1	12.3	6.1
2	905,396	12.6	12.0	4.8
1	1,365,796	12.5	11.9	4.8

Table 1: Comparison of unigram backoff and word clustered backoff for the 1992 DARPA 20k WSJ corpus tested on the feb92m development test set.