N-gram Distribution Based Language Model Adaptation

Jianfeng Gao, Mingjing Li, Kai-Fu Lee

Microsoft Research China

5F. Beijing Sigma Center, No. 49. Zhichun Road, Haidian District, Beijing 100080, P.R.C.

{jfgao, mjli, kfl}@microsoft.com

ABSTRACT

This paper presents two techniques for language model (LM) adaptation. The first aims to build a more general LM. We propose a distribution-based pruning of *n*-gram LMs, where we prune *n*-grams that are likely to be infrequent in a new document. Experimental results show that the distribution-based pruning method performed up to 9% (word perplexity reduction) better than conventional cutoff methods. Moreover, the pruning method results in a more general *n*-gram backoff model, in spite of the domain, style, or temporal bias in the training data.

The second aims to build a more task-specific LM. We propose an *n*-gram distribution adaptation method for LM training. Given a large set of out-of-task training data, called *training set*, and a small set of task-specific training data, called *seed set*, we adapt the LM towards the task by adjusting the *n*-gram distribution in the training set to that in the seed set. Experimental results show non-trivial improvements over conventional methods.

1. INTRODUCTION

Statistical language modeling (SLM) has been successfully applied to many domains such as speech recognition [1], information retrieval [2], and spoken language understanding [3]. In particular, *n*-gram LM has been demonstrated to be highly effective for these domains.

In constructing an *n*-gram LM intended for general text, one is faced with the following two problems.

(1) For the general domain, more training data will always improve a LM. However, as training data size increases, LM size increases, which can lead to models that are too large for practical use. Furthermore, training data is usually biased by its mixture of domain and style, so that the LM will also be biased.

Count cutoff [1] is widely used to prune language models. The method deletes from the LM those *n*-grams that occur infrequently in the training data, assuming that they will be equally infrequent in test data. However, in the real world, training data rarely matches test data perfectly. Worse still, the count cutoff intensifies the bias of the training data.

(2) On the other hand, for specific domains, it usually suffers from sparse-data problems, because large amounts of task-specific data (i.e. training data of specific domain or style, etc) are usually not available.

To deal with the problem, LM is adapted to topic/domain by mixing up LMs that are built for specific and general domains separately [4, 5, 6, 7]. The interpolation weight, which is used to combine the models, is optimized so as to minimize perplexity. However, in the case of combined LM, perplexity has been shown to correlate poorly with recognition performance, i.e. word error rate. Therefore, in many cases, topic adaptation by combining LMs only leads to trivial performance improvements of target systems.

In this paper, we propose a new approach to LM adaptation, which leads to a general LM of limited size whose parameters could be automatically tuned according to the topic/domain of text it is attempting to model.

Two methods of achieving this will be presented.

The first is a distribution-based cutoff method for general LM pruning. This approach estimates if an *n*-gram is “likely to be infrequent in test data”. We developed a criterion for pruning parameters from *n*-gram models, based on the *n-gram distribution* i.e. the probability that an *n*-gram occurs in a document. All *n*-grams with the probability less than a threshold are removed. Experimental results show that the distribution-based pruning method performed up to 9% (word perplexity reduction) better than conventional cutoff methods. Furthermore, the pruning method results in a more general *n*-gram backoff model, in spite of the domain, style and temporal bias of training data.

The second method aims to solve the sparse-data problem of specific domains. We use large amounts of data from other tasks/domains, called *training set*, to improve the task-specific LM. We propose an *n*-gram distribution adaptation method for LM training. Given a set of task-specific training data, called *seed set*, we adapt the LM towards the task by adjusting the *n*-gram distribution in the *training set* to that in the *seed set*. Experiments show non-trivial improvements over conventional methods (e.g. model interpolation approaches).

2. GENERAL LANGUAGE MODELS

2.1 Basic Approach

As described in the previous section, LM pruning is necessary for practical use. But previous cutoff methods are not perfect, and intensify the bias of the training data. For example, if we use newspaper in training, a name like “Lewinsky” may have high frequency in certain years but not others; if we use *Gone with the Wind* in training, “Scarlett O’Hara” will have disproportionately high probability and will not be cutoff.

We propose another approach to pruning. We aim to *keep n-grams that are more likely to occur in a new document*. We therefore propose a new criterion for pruning parameters from n-gram models, based on the *n-gram distribution* i.e. the probability that an *n*-gram will occur in a new document. All *n*-grams with the probability less than a threshold are removed.

We estimate the probability that an *n*-gram occurs in a new document by dividing training data into partitions, called *subunits*, and use a cross-validation-like approach. In the remaining part of this section, we firstly investigate the method for *n-gram distribution modeling*. Then we discuss various ways to divide a training set into subunits. Experiments show that our method outperforms conventional count cutoff by up to 9% word perplexity reduction. Furthermore, it also results in a more *general* n-gram model, in spite of the domain, style, or temporal bias of training data.

For simplicity, in the remaining of this paper, we restrict our discussion to bigram, $p(w_n|w_{n-1})$, which assumes that the probability of a word depends only on the identity of the immediately preceding word. But our approaches extends to any *n*-gram.

2.2 Measure of Generality Probability

In this section, we will discuss in detail how to estimate the probability that a bigram occurs in a new document. For simplicity, we define a document as the subunit of the training corpus. In the next sub-section, we will loosen this constraint.

Term distribution models, which are widely used in IR [8], estimate the probability $P_i(k)$, the proportion of times that of a word w_i appears *k* times in a document. In bigram distribution models, we wish to model the probability that a word pair (w_{i-1}, w_i) occurs in a new document. The probability can be expressed as the measure of the generality of a bigram. Thus, in what follows, it is denoted by $P_{gen}(w_{i-1},w_i)$. The higher the $P_{gen}(w_{i-1},w_i)$ is, for one particular document, the less informative the bigram is, but for all documents, the more general the bigram is.

The standard probabilistic model for the distribution of a certain type of event over units of a fixed size (such as periods of time or volumes of liquid) is the *Poisson distribution* [9]. As stated in [8], the Poisson estimates are good for non-content words, but not for content words. Several improvements over Poisson have been proposed. These include *two-Poisson Model* [10] and Katz's *K mixture model* [11]. The K mixture is better. It is also a simpler distribution that fits empirical distributions of content words as well as non-content words. Therefore, we use K mixture for bigram distribution modelling. According to [11], K mixture model estimates the probability that word w_i appears *k* times in a document as follows:

$$P_i(k) = (1-\alpha)\delta_{k,0} + \frac{\alpha}{\beta+1}\left(\frac{\beta}{\beta+1}\right)^k \qquad (1)$$

where $\delta_{k,0}$=1 iff k=0 and $\delta_{k,0}$=0 otherwise. α and β are parameters that can be fit using the observed mean λ and the observed inverse document frequency *IDF* as follow:

$$\lambda = \frac{cf}{N} \qquad (2)$$

$$IDF = \log \frac{N}{df} \qquad (3)$$

$$\beta = \lambda \times 2^{IDF} - 1 = \frac{cf - df}{df} \qquad (4)$$

$$\alpha = \frac{\lambda}{\beta} \qquad (5)$$

where again, *cf* is the total number of occurrence of word w_i in the collection, *df* is the number of documents in the collection that w_i occurs in, and *N* is the total number of documents.

The bigram distribution model is a variation of the above K mixture model, where we estimate the probability that a word pair (w_{i-1},w_i) , occurs in a document by:

$$P_{gen}(w_{i-1}, w_i) = 1 - \sum_{k=1}^{K} P_i(k) \qquad (6)$$

where *K* is dependent on the size of the subunit, the larger the subunit, the larger the value (in our experiments, we set *K* from 1 to 2), and $P_i(k)$ is the probability of word pair (w_{i-1},w_i) occurs *k* times in a document. $P_i(k)$ is estimated by equation (1), where α , and β are estimated by equations (2) to (5). Accordingly, *cf* is the total number of occurrence of a word pair (w_{i-1},w_i) in the collection, *df* is the number of documents that contain (w_{i-1},w_i), and *N* is the total number of documents.

2.3 Partitioning the training data

In conventional approaches, a document is defined as the subunit of training data for term distribution estimating. But for a very large training corpus that consists of millions of documents, the estimation for the bigram distribution is very time-consuming. To cope with this problem, we use a cluster of documents as the subunit. As the number of clusters can be controlled, we can define an efficient computation method, and optimise the clustering algorithm. For example, in experiments stated in [12], documents are clustered in three ways: by similar domain, style, or time.

2.4 Results

Figure 1 shows the results when we define a document as the subunit. We used approximately 450 million characters of People's Daily training data (1996), which consists of 39,708 documents. The test data consists of 15 million characters that have been proofread and balanced among domain, style and time. As shown in figure 1, although as the size of LM is decreased, the perplexity rises sharply, the models created with the bigram distribution based pruning have consistently lower perplexity values than for the count cutoff method. Further analysis shows that a lot of domain, style or time specific bigrams are pruned using our pruning method. Thus our pruning method results in a more general *n*-gram model, which resists to domain, style or temporal bias of training data.

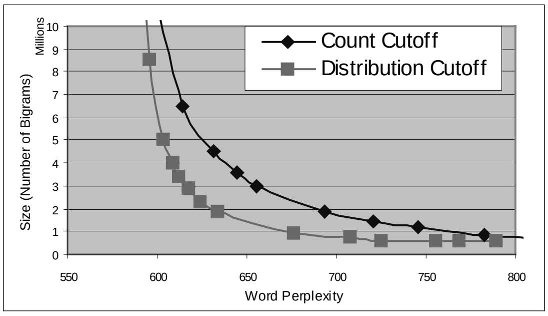

Figure 1: Word perplexity comparison of cutoff pruning and distribution based bigram pruning.

3. TASK-SPECIFIC LANGUAGE MODELS

3.1 Basic Approach

As described in the previous section, previous LM combination methods assume that perplexity reduction will definitely lead to performance improvements. Unfortunately, in the case of combined LM, it is not true.

We find that *it is the n-gram distribution that characterizes the task-specific training data*. Therefore, in this paper, we investigate an alternative approach to using out-of-task/domain training data. We propose an approach based on *n*-gram distribution adaptation for LM training. Given a large set of out-of-task/domain training data, called *training set*, and a small set of task-specific training data, called *seed set*, we adapt the LM towards the task by adjusting the *n*-gram distribution in the training set to that in the seed set.

Instead of combining bigram models built on training set and seed set, respectively. We directly combine bigram counts *C(w_1,w_2)* with an adaptation weight *W(w_1, w_2)* in the form of

$$C(w_1, w_2) = \sum_i W_i(w_1, w_2) \times C_i(w_1, w_2) \tag{7}$$

where *Wi(w_1, w_2)* is the adaptation weight of the *i*th training set, it is estimated by

$$W_i(w_1, w_2) = \log \left(\frac{p(w_1, w_2)}{p_i(w_1, w_2)} \right)^{\alpha} \tag{8}$$

where α is the adaptation coefficient, and *p(w_1,w_2)* is the frequency of the bigram *(w_1,w_2)* in the seed set, and *p_i(w_1,w_2)* is the frequency of the bigram *(w_1,w_2)* in the *i*th training set, which is estimated by

$$p_i(w_1, w_2) = \frac{C_i(w_1, w_2)}{\sum_{w_1, w_2} C_i(w_1, w_2)} \tag{9}$$

The key points of the *n*-gram distribution adaptation are the selection of α, and the seed set, which will be described below.

3.2 Seed Set and Overfitting

The task-specific training data is usually defined as the seed set. The seed set is not large enough, so it almost never covers real task data. This can lead to the overfitting problem.

To avoid overfitting, we divide the seed set into partitions, and use a cross-validation-like approach. In what follows, we will describe what we do in our experiments.

We randomly divide the task-specific training data into 5 partitions, each of the same size. Each time, we pick one partition as a *testing set* and combine the other 4 partitions as a *seed set*. Then we set α from 0.0 to 2.5, and combine bigram counts by equations (7) to (9). Finally, a series of bigram models are estimated. As shown in figure 2, as α increases, the perplexity to the seed set is reduced, and the perplexity to the testing set is firstly reduced and then raised sharply. Thus, we pick the value of α, where the perplexity to the testing set just begins to rise. We call the value at this point of α.the *critical value* of α. We redo the above experiment by defining different pairs of testing set and seed set, and obtained the average *critical value* of α.

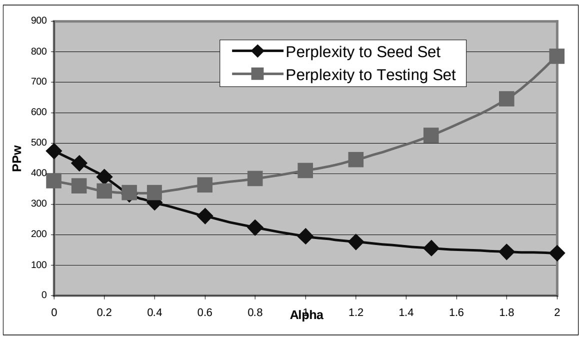

Figure 2: N-gram distribution adaptation for LM training

3.3 Training Data Optimization

In applying an SLM, more training data will usually improve a LM. While it is possible to collect large amount of training data (such as from websites), the majority of the data is likely to be irrelevant to the given task. Thus a method of training data optimization such as suggested in [7] can be used to select relevant text materials to build a more task-specific LM. Moreover, there is never infinite memory, the method is also subject to a memory constraints.

Our approach here is to take the available task-specific data, called the *seed set*, and a large variety of data (e.g. data collected from the web), called the *training set*, and train a language model which not only satisfies the memory constraint but also has the best performance. The basic approach involves following steps:

1. Take the large training set, and divided it up into units, so that we can decide whether to keep each unit, and how much to trust each unit.
2. Assign a score to each unit using perplexity as the metric. We train a LM from our seed set, and measure each training data unit's test-set perplexity against this LM.
3. Add the best 5% or more units of the training set to the seed set, train a LM on the seed set while keeping its size to meet the memory constraint by a relative entropy based cut-off method.
4. Repeat step 3 until the improvement of perplexity of the LM is less than a pre-set threshold.

As shown in the next sub-section, training set optimization give improved perplexity and recognition performance over more simplistic methods of using out-of-domain data.

3.4 Results

Perplexity and recognition experiments were run on the input method editor (IME – a software layer that converts keystrokes into Chinese characters) task. The in-domain training data includes 40 million characters from available application documents of IME (IME). The out-of-domain training data includes 300 million characters of variety text collected from Chinese websites (WEB). The test data consists of 15 million characters that have been proofread.

The recognition results are obtained using the system of pinyin-to-character conversion. This is a similar problem to speech recognition. The system we use is MSPY2.0, developed by IME group in Microsoft, which is one of the best products and delivers the best accuracy today in spite of minimal memory usage.

Table 1 compares our *n*-gram distribution based LM adaptation and training data optimization techniques described above, and provides the results obtained using only an IME trigram model (trained from IME data only) as a baseline. The first column contains the training data components, the second column shows the techniques we use, the third and the fourth columns show the word perplexity (PPW) and the word error rate (WER) of pinyin-to-character conversion, respectively, and the fifth column shows the percentage reduction in word error rate (RED-WER) from the baseline.

Row 3 shows that simply adding out-of-domain training data, called brute-force (BF) technique, results in 8.08 % word error rate reduction. When we simply interpolate the IME trigram and the LM trained from out-of-domain training data, called single interpolation (SI) scheme, a slight improvement, say 10.5% word error rate reduction, can be obtained. Most previous methods [4, 5, 6] are based on BF and SI. We then combine BF and SI with the training data optimization method described in the last sub-section. They are denoted by OBF and OSI in table 1, respectively. Further improvements are obtained, as shown in rows 5 and 6. Row 7 shows that the best results are obtained using the *n*-gram distribution based LM adaptation (DBA) method. Moreover, that fact that perplexity decreases are associated with the worse recognition results suggests that test set perplexity may not be a good criterion for estimating model interpolation weights.

LM Training	Technique	PPW	WER	RED-WER
IME	Baseline	645.24	7.05 %	
IME+WEB	BF	432.71	6.48 %	8.08 %
IME+WEB	SI	388.34	6.31 %	10.50 %
IME+WEB	OBF	411.77	6.21 %	11.91 %
IME+WEB	OSI	382.46	6.26 %	11.21 %
IME+WEB	DBA	389.87	5.80 %	17.73 %

Table 1: Perplexity and recognition results.

4. CONCLUSION

This paper presents two techniques for LM adaptation. The first aims to build a more general LM. We propose a distribution-based pruning of *n*-gram LMs, where we prune *n*-grams that are likely to be infrequent in a new document. Experimental results show that the distribution-based pruning method performed up to 9% (word perplexity reduction) better than conventional cutoff methods. Moreover, the pruning method results in a more general *n*-gram backoff model, in spite of the domain, style, or temporal bias in the training data.

The second aims to build a more task-specific LM. We propose an *n*-gram distribution adaptation method for LM training. Given a large set of out-of-task training data, called *training set*, and a small set of task-specific training data, called *seed set*, we adapt the LM towards the task by adjusting the *n*-gram distribution in the training set to that in the seed set. We present a method to avoid overfitting. A method of training data optimization is also described briefly. Experiments show non-trivial improvements over conventional methods (e.g. model interpolation approaches).

5. ACKNOWLEDGEMENT

We would like to thank Zheng Chen, Ming Zhou, Eric Chang, Chang-Ning Huang and other colleagues from Microsoft Research, Jian-Yun Nie from the University of Montreal, Canada, Charles Ling from the University of Western Ontario, Canada, and Lee-Feng Chien from Academia Sinica, Taiwan, for their help in developing the ideas and implementation in this paper.

6. REFERENCE

1. F. Jelinek, “Self-organized language modeling for speech recognition”, in *Readings in Speech Recognition*, A. Waibel and K.F. Lee, eds., Morgan-Kaufmann, San Mateo, CA, 1990, pp. 450-506.

2. D. Miller, T. Leek, R. M. Schwartz, “A hidden Markov model information retrieval system”, in Proc. 22nd International Conference on Research and Development in Information Retrieval, Berkeley, CA, 1999, pp. 214-221.

3. V.W. Zue, “Navigating the information superhighway using spoken language interfaces”, IEEE Expert, vol. 10, no. 5, pp. 39-43, October, 1995

4. R. Iyer, M. Ostendorf, H. Gish, “Using Out-of-Domain Data to Improve In-Domain Language Models”, IEEE SIGNAL PROCESSING LETTERS, Vol. 4. No. 8. Aug, 1997

5. P. Clarkson and A. Robinson. “Language model adaptation using mixtures and an exponentially decaying cache”, In *Proc. ICASSP-97*, 1997.

6. K. Seymore, R. Rosenfeld, “Using story topics for language model adaptation”, in *Proc. ICASSP-97*, 1997

7. J. Gao, H. F. Wang, M. Li, and K. F. Lee. “A Unified Approach to Statistical Language Modeling for Chinese”. In *IEEE, ICASPP2000,* 2000

8. C. D. Manning, and H. Schütze, “Foundations of Statistical Natural Language Processing”, The MIT Press, 1999.

9. M. Mood, A. G. Franklin, and C. B. Duane, “Introduction to the theory of statistics”, New York: McGraw-Hill, 3rd edition, 1974.

10. S. Harter, “A probabilistic approach to automatic keyword indexing: Part II. An algorithm for probabilistic indexing”, Journal of the American Society for Information Science, 1975(26): 280-289

11. S. M. Katz, “Distribution of content words and phrases in text and language modeling”, Natural Language Engineering, 1996(2): 15-59

12. J. Gao, K. F. Lee, “Distribution-Based Pruning of Backoff Language Models” In *Proceedings of The Annual Meeting of the ACL*, Hong Kong, 3 - 6 October 2000.