Modeling Long Distance Dependence in Language: Topic Mixtures vs. Dynamic Cache Models

R. Iyer M. Ostendorf

rukmini, mo@bu.edu Electrical and Computer Engineering Department Boston University, Boston, MA 02215, USA

ABSTRACT

In this paper, we investigate a new statistical language model which captures topic-related dependencies of words within and across sentences. First, we develop a sentence-level mixture language model that takes advantage of the topic constraints in a sentence or article. Second, we introduce topic-dependent dynamic cache adaptation techniques in the framework of the mixture model. Experiments with the static (or unadapted) mixture model on the 1994 WSJ task indicated a 21% reduction in perplexity and a 3-4% improvement in recognition accuracy over a general n-gram model. The static mixture model also improved recognition performance over an adapted n-gram model. Mixture adaptation techniques contributed a further 14% reduction in perplexity and a small improvement in recognition accuracy.

1. Introduction

Statistical language models, which model the probability of different word sequences, are an integral part of state-of-the-art speech recognizers. The most commonly used statistical language modeling technique, also referred to as n-gram language modeling, considers the word sequence w_1, w_2, \ldots, w_T to be a Markov process with probability

$$P(w_1, w_2, \dots, w_T) = \prod_{i=1}^{T+1} P(w_i | w_{i-1}, \dots, w_{i-n+1})$$
 (1)

where w_0 and w_{T+1} are sentence boundary markers and n refers to the order of the Markov process (typically 2 or 3, a bigram or trigram language model respectively). For notational simplicity in the discussion, we use the bigram representation; however, all experiments in this paper use trigram models.

Although quite powerful given their simplicity, *n*-gram models are constrained in their inability to take advantage of dependencies longer than *n*. One approach to overcome this limitation is to use dynamic cache language models [1, 2, 3], which model task-level dependencies by increasing the likelihood of a word given that it has been observed previously. However, cache models do not account for dependencies within a sentence. Context-free grammars [4, 5] could account for grammatical dependencies within a sentence; however, it is costly to build task-specific grammars. Trigger language models [6] capture dependencies within a sentence but are computationally very expensive while training the models.

Our approach to representing long term dependence attempts to address both task-level and sentence-level dependencies, while using a simple model. We investigate a sentence-level mixture of m component language models [7], each of which can be identified with the n-gram statistics of a specific topic or a broad class of sentences. Though this approach is similar to those that use mixtures at the n-gram level [8, 9], it differs in the use of mixtures at the sentence level and automatic clustering to define the mixtures. Specifically, the probability of a word sequence w_1, \ldots, w_T is given as,

$$P(w_1, \dots, w_T) = \sum_{k=1}^{m} \lambda_k \left[\prod_{i=1}^{T+1} P_k(w_i | w_{i-1}) \right]$$
 (2)

where λ_k are the mixture weights and $P_k()$ is the n-gram model for the k-th class. The sentence-level mixture can be used either as a static or dynamic model, easily leveraging existing techniques for adaptive language modeling. The recognition search cost of a sentence-level mixture model is potentially high, but in an N-best rescoring framework, the additional cost of the language model is minimal.

The general framework and mechanism for designing the mixture language model will be described in Section 2, including descriptions of automatic topic clustering and robust parameter estimation techniques. Next, Section 3 presents an extension of the mixture model to incorporate dynamic adaptation techniques. Section 4 describes the experimental paradigm and results, and finally, Section 5, concludes with a discussion of possible extensions of mixture language models.

2. Mixture Model Framework

Training the model given in Equation 2 requires that we address two main issues: automatic clustering to handle data not explicitly marked for topic dependence, and robust parameter estimation. Our solution to these problems is described below.

2.1. Clustering

Topics or natural groupings of data can be specified by hand if text labels are available or determined automatically, which is the approach taken here since we used a corpus that does not have topic labels associated with the data. Agglomerative clustering is used to partition the training data into the desired number of clusters/topics [7]. Clustering is performed at the article level to reduce computation, relying on the assumption that an entire article comes

from a single topic. Starting with single-article clusters, clusters are progressively grouped by computing the similarity and grouping the most similar two clusters. The similarity measure is based on the combination of inverse document frequencies [10], specifically

$$S_{ij} = \sum_{w \in A_i \cap A_j} \frac{N_{ij}}{|A^w|} \times \frac{1}{|A_i||A_j|}$$
 (3)

where $|A_i|$ is the number of unique words in article i, $|A^w|$ is the number of articles containing the word w and

$$N_{ij} = \sqrt{\frac{N_i + N_j}{N_i \times N_j}} \tag{4}$$

is a normalization factor with N_i being the number of articles in cluster i. The normalization factor is used to avoid the tendency for small clusters to group with one large cluster rather than with other small clusters. An advantage of the inverse document frequency measure is that high frequency words, such as function words, are automatically discounted.

2.2. Robust Parameter Estimation

In parameter estimation, two important issues are addressed. First, the initial topic-dependent clusters may not be optimal for estimating the n-gram parameters of the m-component models, in which case better results can be obtained by iterative re-estimation. Second, we need to account for articles or sentences that do not belong to any of the topics as well as consider the sparse data problems caused by the fragmentation of the training data, which we address by using double mixtures, one at the sentence-level and the other at the n-gram level.

Iterative Topic Model Re-Estimation Initial estimates for each of the component n-gram models are based on the partitions of the training data obtained by the clustering procedure described earlier. It is essential that some back-off smoothing technique be used in the component models to account for n-grams not observed in training; here the Witten-Bell technique [11] is used. We iteratively re-estimate these parameters, since some sentences belonging to one topic according to the clustering metric might be more likely in another topic according to the n-gram models.

Initially, we developed a *Viterbi-style* training technique that iteratively partitions the data by topic and re-estimates the topic-dependent parameters [7]. Here, we instead use the Expectation-Maximization (EM) algorithm, where we assign the same sentence to different topics weighted by the likelihood of that particular topic. The EM algorithm is complicated for language model training, because the parameter estimates must incorporate some sort of back-off to avoid zero probabilities. The simplicity of the Witten-Bell back-off scheme facilitates the integration of the back-off directly into the EM algorithm as a constraint in re-estimation.

In each EM iteration, we first compute the likelihood of every sentence in the training corpus given each of the m topics. For a bigram mixture model, the likelihood that the i^{th} training sentence belongs

to the j^{th} topic in the p^{th} iteration is

$$\hat{z}_{ij}^{(p)} = \frac{P^{(p)}(y_i|z_j)P^{(p)}(z_j)}{\sum_{j=1}^m P^{(p)}(y_i|z_j)P^{(p)}(z_j)}$$
(5)

where z_j is the j^{th} class and y_i is the i^{th} sentence. The bigram probability for the jth topic-model is then re-computed as

$$P_i(w_c|w_b) = (1 - \phi_b^j) P_i^{ML}(w_c|w_b) + \phi_b^j P_i(w_c),$$
 (6)

where ϕ_b^j is the unigram back-off mass

$$\phi_b^j = \frac{\sum_{q} \frac{\sum_{i=1}^n n_{bq}^i \hat{z}_{ij}^{(p)}}{\sum_{i=1}^n n_{bq}^i}}{\sum_{i=1}^n n_{bq}^i \hat{z}_{ij}^{(p)}}, \qquad (7)$$

$$\sum_{q} \sum_{i=1}^n n_{bq}^i \hat{z}_{ij}^{(p)} + \sum_{q} \frac{\sum_{i=1}^n n_{bq}^i \hat{z}_{ij}^{(p)}}{\sum_{i=1}^n n_{bq}^i},$$

n represents the number of training sentences, n_{bc}^i represents the number of occurrences of the bigram $< w_b, w_c >$ in the i^{th} sentence and $P_i^{ML}(w_c|w_b)$ is given as

$$P_j^{ML}(w_c|w_b) = \frac{\sum_{i=1}^n \hat{z}_{ij}^{(p)} n_{bc}^i}{\sum_{a} \sum_{i=1}^n n_{ba}^i \hat{z}_{ij}^{(p)}}.$$
 (8)

Equations 6-8 give the model parameter estimates for the next iteration. The derivation of this solution is given in [12].

Topic Model Smoothing For smoothing the n-gram estimates of the component models to handle fragmentation of the training data, we interpolate with the general model at the n-gram level. We deal with the problem of "non-topic" sentences by including a general model in addition to the m component models at the sentence level. Specifically, the model is now given by

$$P(w_1, ..., w_T) = \sum_{k=1}^{m,G} \lambda_k \prod_{i=1}^{T+1} [\theta_k P_k(w_i | w_{i-1}) + (1 - \theta_k) P_G(w_i | w_{i-1})], \quad (9)$$

where P_G refers to a general model trained on all the available training data, λ_k are the sentence-level mixture weights, and θ_k are the n-gram level mixture weights.

The two sets of weights, λ_k and θ_k , are estimated separately using a held-out data set. The sentences in the held-out data set are first labeled according to their most likely topic and thus the data is split into m clusters. A simple maximum likelihood criterion is used to estimate the θ 's iteratively beginning with uniform weights for each θ . After the component model smoothing factors have been estimated, the λ 's are estimated at the sentence-level using an analogous algorithm over the entire data.

3. Dynamic Adaptation

The sentence-level mixture model defined in the previous section has static parameters, i.e. the n-gram estimates are not updated as we observe new sentences. However, it is easy to extend the dynamic modeling techniques of cache language models to our mixture language model, as shown next.

3.1. Dynamic *n*-gram Cache Adaptation

In dynamic cache adaptation, the entire partially observed document is cached and used to build an n-gram model. The cache probabilities are linearly interpolated with the static n-gram probabilities, as follows,

$$P(w_1, \dots, w_T) = \prod_{i=1}^{T+1} (\mu^s P^s(w_i|w_{i-1}) + \mu^c P^c(w_i|w_{i-1}))$$
 (10)

where P^s represents the smoothed static model parameters, P^c represents the cache model parameters, and μ^s and μ^c represent the weights for the static and cache models respectively. The interpolation weights μ^s and μ^c for the cache language models (Equation 10) were estimated on the development test data to minimize recognition word error rate in a series of experiments.

The important issues to be resolved in dynamic language modeling with a cache are the definition of the cache and the mechanism for choosing the interpolation weights. We experimented with two cache models. Since the function words are well estimated due to their high frequency in the training data, we considered a content word unigram cache. This is similar to the approach suggested by Rosenfeld [3] where only *rare* words observed in a document are cached, *rare* being defined by the frequency of the word in the training corpus. We observed that defining "rare" as a content word alleviated the problem of deciding a threshold frequency below which a word is considered *rare*, as well as gave us small but consistent improvements in performance over the frequency-based rare-word cache. We also worked with a conditional bigram/trigram cache [3], which is used in addition to the content word unigram cache.

3.2. Adaptation in Mixture Framework

To extend cache-based n-gram adaptation to the mixture model, we maintain separate counts for all the component models, including the general model. A sentence is weighted according to its likelihood given each of the component models. For the k^{th} model, the likelihood for sentence w_1,\ldots,w_T is given by Equation 5. Fractional counts are then assigned to each topic according to their relative likelihoods, which allows all the topic-models to take advantage of the partially observed document. The equation for the adapted mixture model incorporating component-caches is a combination of Equation 2 and Equation 10.

We did not use dynamic mixture weight adaptation [8], though we developed a recursive solution to this problem in [12], since initial experiments with this algorithm indicated no significant improvement in terms of perplexity or word error rate.

4. Experiments

4.1. Paradigm

The language model training corpus, referred to as the North American Business (NAB) corpus, encompasses the period from 1988-1994 and consists of approximately 230 million words of articles in the SGML format. We also used 1 million words of set-aside test

data from the NAB corpus to estimate the sentence-level and the n-gram-level mixture weights. The lexicon for all the experiments was a 46K vocabulary provided by BBN.

The results are based on the BU Stochastic Segment Model (SSM) recognition system [13], using the N-best rescoring formalism with N-best hypotheses generated by the BBN Byblos System, a speaker independent HMM system. In this formalism, the top N sentence hypotheses (in our case N=100) are rescored by the SSM and the mixture LM, and a weighted combination of scores from different knowledge sources is used to re-rank the hypotheses. The top ranking hypothesis is used as the recognized output. The weights for re-combination are estimated on the development test data (1994 H1 development test) and held fixed for the evaluation test set.

We report recognition word error rates and perplexity numbers on the 1994 ARPA development and evaluation test sets, referred to as the Hub 1 (H1) condition [13]. We test on both the H1-P0 and the H1-C2 conditions. The N-best used in both these conditions are the same; the difference lies in using unadapted language models vs. adapted language models. The adaptation is supervised, i.e. the true transcriptions are available. The *n*-gram level and sentence-level mixture weights, as well as the cache, are reset at the end of a session (typically 15 sentences). Unless stated, no BBN knowledge sources are used in these experiments.

4.2. Results

In all experiments described here, a 5-component sentence mixture is used. Details of our exploratory work regarding different similarity measures and training techniques are in [12], [7]. We assessed the usefulness of the mixture language model using two criteria; recognition word error rate and perplexity. Table 1 compares four different language models in terms of perplexity on both the development and evaluation test sets. The unadapted mixture

Table 1: Perplexity: 1994 NAB 46K development and evaluation test sets.

Test	Adaptation	Trigram Model	5-component mixture model
Dev	No	211	165
Dev	Yes	171	141
Eval	No	210	175
Eval	Yes	175	145

model reduces perplexity by 22% over the unadapted trigram language model. The mixture model also provides a small perplexity improvement over the adapted trigram language model on the development test set. Incorporating dynamic cache modeling techniques in the mixture framework leads to a further improvement in the mixture model of 14.5%.

Table 2 gives the recognition performance of the four different language models; namely the adapted and unadapted general and mixture trigram models. The recognition results are consistent with the

Table 2: Recognition word error rate on the 1994 NAB development and evaluation test sets.

Test	Adaptation	Trigram model	5-component mixture model
Dev	No	10.5%	10.2%
Dev	Yes	10.3%	10.2%
Eval	No	11.5%	11.0%
Eval	Yes	11.1%	10.8%

improvements in perplexity. The mixture model gives a 3-4% improvement in recognition accuracy over the general model. Adaptation in the mixture framework gives a small but significant gain on the evaluation test set, although no gain was observed on the development test set.

In addition to working with the BU acoustic models, confirming that the mixture model is more powerful than supervised adaptation for short sessions. We also performed recognition experiments combining the mixture model with the BBN baseline acoustic model. We obtained a best case performance of 8.9% word error rate on the development test set and 10.6% on the evaluation test set, as compared to 9.3% and 11.1% respectively, using an adapted general model, confirming that the mixture model is more powerful than supervised adaptation for short sessions.

5. Conclusions

The sentence-level mixture model has the potential to capture long range within-sentence effects and topic dependent effects across sentences using a simple variation of the n-gram approach. To implement the mixture model, we developed an automatic clustering algorithm to classify text, introduced two levels of mixture models for smoothing, and developed an EM algorithm for n-gram probability estimation with a Witten-Bell back-off constraint. Our experiments show that the static (unadapted) topic-dependent mixture model outperforms the standard n-gram model, and that the static mixture model improves recognition performance over a dynamically adapted n-gram model. Incorporating cache models in the mixture framework provides us with an additional reduction in both perplexity and word error rate.

This work can be extended in several ways. One approach to overcome fragmentation problems in very sparse-data domains could include using an *n*-gram part-of-speech sequence model as the base for all component models and topic-dependent word likelihoods given the part-of-speech labels, a natural extension of [2], assuming that the part-of-speech sequence probabilities are not topic-dependent and can be based on the entire training data. The simple static mixture language model can also be useful in applications other than continuous speech transcription. For example, topic-dependent models could be used for topic spotting. The mixture framework could also be extended to use speaker/gender style or goal-related dependencies.

In summary, we present a new language model that gives significant gains in recognition performance over unadapted as well as adapted n-gram models. At the same time the model is a simple variation

of *n*-gram techniques and hence other language modeling advances can be easily incorporated in this framework.

Acknowledgments

This work was supported jointly by ARPA and ONR on grant ONR, number ONR-N00014-92-J-1778. We gratefully acknowledge the cooperation of several researchers at BBN, who provided the N-best hypotheses used in our recognition experiments.

6. REFERENCES

- F. Jelinek, B. Merialdo, S. Roukos and M. Strauss, "A Dynamic LM for Speech Recognition", *Proc. ARPA Workshop on Speech* and Natural Language, pp. 293-295, 1991.
- 2. R. Kuhn and R. de Mori, "A Cache Based Natural Language Model for Speech Recognition", *IEEE Transactions PAMI*, Vol. 14, pp. 570-583, 1992.
- 3. R. Rosenfeld, "A Hybrid Approach To Adaptive Statistical Language Modeling", *Proc. ARPA Workshop on Human Language Technology*, pp. 76-87, 1994.
- J. H. Wright, G. J. F. Jones and E. N. Wrigley, "Hybrid Grammar Bigram Speech Recognizer System with First-Order Dependence Model", *Proc. ICASSP*, Vol. I, pp. 169-171, 1992.
- M. Meteer and J. R. Rohlicek, "Statistical Language Modeling Combining N-gram and Context Free Grammars", *Proc. ICASSP*, Vol. II, pp. 37-40, 1993.
- R. Lau, R. Rosenfeld and S. Roukos, "Trigger-Based Language Models: a Maximum Entropy Approach", *Proc. ICASSP*, Vol II, pp. 45-48, April 1993.
- R. Iyer, M. Ostendorf and R. Rohlicek, "An Improved Language Model Using a Mixture of Markov Components", Proc. ARPA Workshop on Human Language Technology, pp. 82-87, 1994.
- R. Kneser and V. Steinbiss, "On the Dynamic Adaptation of Stochastic LM", Proc. ICASSP, Vol. 2, pp. 586-589, 1993.
- L. Bahl et al., "The IBM Large Vocabulary Continuous Speech Recognition System for the ARPA NAB News Task," Proc. ARPA Workshop on Spoken Language Technology, pp. 121-126, 1995.
- 10. S. Sekine, "Automatic Sublanguage Identification for a New Text", *Second Annual Workshop on Very Large Corpora, Kyoto, Japan*, pp.109-120, August 1994.
- P. Placeway, R. Schwartz, P. Fung and L. Nguyen "Estimation of Powerful LM from Small and Large Corpora", *Proc. ICASSP*, pp. 33-36 Vol. 2, April 1993.
- 12. R. Iyer, *Language Modeling with Sentence-Level Mixtures*, Boston University M. S. Thesis, 1994. (anonymous ftp to raven.bu.edu, in the pub/reports directory)
- M. Ostendorf, F. Richardson, R. Iyer, A. Kannan, O. Ronen and R. Bates, "The 1994 BU NAB News Benchmark System", *Proc.* ARPA Workshop on Spoken Language Technology, pp. 139-142, 1995.