IMPLEMENTING AND IMPROVING MMIE TRAINING IN SPHINXTRAIN

Long Qin and Alexander Rudnicky

Carnegie Mellon University
Language Technologies Institute
5000 Forbes Avenue, Pittsburgh, PA 15213, USA
{lqin, air}@cs.cmu.edu

ABSTRACT

Discriminative training schemes, such as Maximum Mutual Information Estimation (MMIE), have been used to improve the accuracy of speech recognition systems trained using Maximum Likelihood Estimation (MLE). In this paper, we present the implementation details of MMIE training in SphinxTrain and baseline results for MMIE training on the Wall Street Journal (WSJ) SI84 and SI284 data sets. This paper also introduces an efficient lattice pruning technique that both speeds up the process and increases the impact of MMIE training on recognition accuracy. The proposed pruning technique, based on posterior probability pruning, is shown to provide better performance than MMIE using standard pruning techniques.

Index Terms— SphinxTrain, MMIE training, word lattice, lattice pruning

1. INTRODUCTION

CMU Sphinx is a popular open source speech recognition system [1]. It is currently used by researchers and developers in many locations world-wide, including universities, research institutions and in industry. SphinxTrain is the acoustic model trainer component which provides a variety of tools for creating acoustic models. To date, SphinxTrain did not include an implementation of discriminative training, such as Maximum Mutual Information Estimation (MMIE) training [2].

MMIE training has been used to provide better performance than maximum likelihood estimation (MLE) in speech recognition systems. Naively applied, MMIE attempts to maximize the posterior probability of the correct word sequence given all possible word sequences, but this requires a prohibitive amount of computation to estimate confusable hypotheses and perform parameter estimation and so is impractical. The computational bottleneck can be eliminated by performing lattice-based MMIE training [3]. In such lattice-based training framework, the word lattice, which contains a set of word hypotheses with boundary times and transitions, is used as a compact representation of competing hypotheses [4]. And then parameter optimization is conducted using the extended Baum-Welch (EBW) algorithm [5].

Sphinx lattices contain many individual hypotheses with the same word, differentiated by entry or exit times. When doing MMIE training on such lattices, most of time a model just competes with itself but not other confusable models. The end result is that acoustic modeling can't fully benefit from MMIE training. We investigated the effect of MMIE training on pruned lattices, which contain fewer duplicate and unnecessary word hypotheses. One of the most effective lattice pruning methods is beam pruning during decoding [6]. There are also many other lattice pruning methods that are usually used to prune lattices as part of multi-pass decoding. In those methods, a word lattice is usually converted to a word graph and all time alignment information is discarded. This includes finite state automata (FSA) determination and minimization [7] and the confusion networks approach [8].

Unfortunately the above lattice pruning methods are not appropriate for the purpose of lattice pruning for discriminative training. The word lattice used in MMIE training is quite different from the lattice used for decoding: 1) normally a unigram language model is used for calculating the language model score for word hypotheses [9][10], 2) the time alignment information needs to be preserved for EBW computation 3) the Gaussian occupation count is weighted by the posterior probability of the word hypothesis, therefore beam pruning which only considers the forward likelihood is insufficient. We propose a new lattice pruning method that keeps the boundary times of word hypotheses in the lattice and prunes unnecessary word hypotheses by considering their posterior probabilities. The posterior probability of a word hypothesis is calculated in the same way as in EBW by doing the forward-backward computation on the lattice. Using this method, word hypotheses which do not contribute much to Gaussian count accumulation are removed from the lattice. Furthermore, in our posterior probability pruning method, we also try to reduce the number of duplicate word hypotheses. This allows us to get greater improvement from MMIE training on such pruned lattices. Another advantage of MMIE training on pruned lattices is that it reduces computation in both the generation of word lattices and in the EBW computation. Such improvements make particular sense for SphinxTrain which is intended to be useful even in environments with relatively modest computational resources. A similar idea of performing discriminative training on pruned lattice to save computation was mentioned in [11]. However, in this paper, we try to both reduce the computation cost and improve the recognition accuracy.

The remainder of this paper is organized as follows. In section 2, implementation details of MMIE training in SphinxTrain are provided. Then Section 3 describes the beam pruning method. The details of the posterior probability lattice pruning algorithm are presented in Section 4. And in Section 5 and 6 the experiment setup and results are provided. Concluding remarks are provided in Section 7.

2. MAXIMUM MUTUAL INFORMATION ESTIMATION

As an alternative of the MLE training of HMMs, MMIE training attempts to optimize the correctness of a model by formulating an objective function that penalizes the confusable models relative to

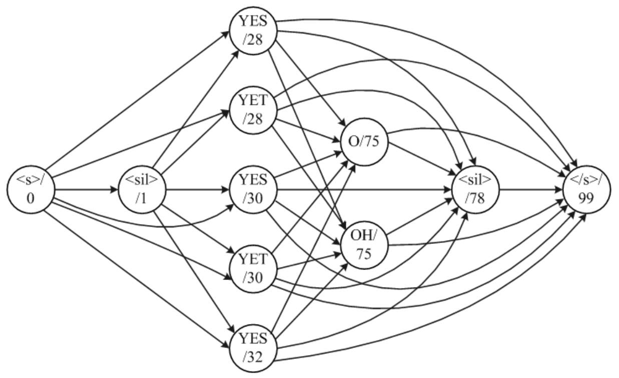

Fig. 1. A lattice corresponding to "<s> YES </s>"

the true model,

$$F(\lambda) = \sum_{r=1}^{R} \log \frac{P_\lambda(O_r|M_{s_r})P(S_r)}{\sum_s P_\lambda(O_r|M_s)P(S)}, \tag{1}$$

where λ represents the acoustic model parameters, O_r are the training utterances, M_s is the model corresponding to a word sequence s, and s_r is the correct transcription for the rth utterance, while $P(S)$ is a weakened language model such as a unigram language model. Thus, MMIE tries to maximize the likelihood of the correct transcription and simultaneously minimize the likelihood of the competing word hypotheses.

The MMIE objective function can be optimized using the EBW algorithm, and the mean and variance of a particular dimension of the Gaussian mixture component m for state j can be re-estimated as follows

$$\hat{\mu}_{j,m} = \frac{\{\theta_{j,m}^{num}(O) - \theta_{j,m}^{den}(O)\} + D_{jm}\mu_{j,m}}{\{\gamma_{j,m}^{num} - \gamma_{j,m}^{den}\} + D_{j,m}} \tag{2}$$

$$\hat{\sigma}_{j,m}^2 = \frac{\{\theta_{j,m}^{num}(O^2) - \theta_{j,m}^{den}(O^2)\} + D_{j,m}(\sigma_{j,m}^2 + \mu_{j,m}^2)}{\{\gamma_{j,m}^{num} - \gamma_{j,m}^{den}\} + D_{j,m}} - \hat{\mu}_{j,m}^2, \tag{3}$$

where $r_{j,m}$ is the occupation count of the Gaussian mixture component, and $\theta_{j,m}(O)$ and $\theta_{j,m}(O^2)$ are the standard weighted sums over feature $x(t)$ and $x(t)^2$ respectively. The Gaussian-specific learning rate constant $D_{j,m}$ is used to control the speed of the parameter update and is set to be large enough to ensure that all Gaussian variances remain positive. The superscripts num and den refer to statistics collected on the numerator lattices and denominator lattices, respectively. The numerator lattice can be generated by aligning the acoustic data against a network of HMMs built according to the correct transcription. The denominator lattice corresponds to all possible competing word sequences and is a by-product of the recognition process over the training utterances. Figure 1 gives an example of a denominator lattice generated using the Sphinx III decoder. In this lattice, each node is associated with a word and its entry time. An arc is the transition from one node to another. In our experiment, to simplify the EBW implementation, we converted the numerator lattices and denominator lattices into a simpler format in which each element contains a word hypothesis and the indexes for preceding and succeeding hypotheses. In this simplified format, word hypotheses are sorted by their entry time.

The above update equations require accumulating statistics for the numerator and denominator HMMs by performing a lattice-based forward-backward computation. Assuming that for a word lattice, we have $q = 1...Q$ arcs sorted in time. And we have already calculated the pre-scaled language model score $lm(q)$ and acoustic likelihood $ac(q)$ for each arc. In this paper, the language model score $lm(q)$ comes from a unigram language model, while the acoustic likelihood $ac(q)$ is estimated using Viterbi. Then the posterior probability of each arc $\gamma(q)$ can be computed as:

1. Initialize $\alpha(1) = \beta(Q) = 1.0$
2. for arc $q = 1...Q$, for arc r preceding q,
   $\alpha(q) += \alpha(r) * ac(q) * lm(q)$
3. for arc $q = Q...1$, for arc r following q,
   $\beta(q) += \beta(r) * ac(q) * lm(q)$
4. for arc $q = 1...Q$,
   $\gamma(q) = \alpha(q) * \beta(q)/(ac(q) * lm(q) * \beta(1))$

3. BEAM PRUNING

Beam pruning is a very effective way to do lattice pruning. It is applied during the recognition process over the training data. During decoding, the decoder only expands the HMM states that have accumulated likelihoods which are greater than a beam width multiplied by the best accumulated likelihood so far. So we can eliminate those states that have very low likelihoods and end up with a very small lattice. But in MMIE training, where the Gaussian mixture occupation count is weighted by the posterior probabilities of word hypotheses, pruning only by accumulated likelihood is not appropriate. Also in beam pruning, when a state is pruned, all word sequences ending with that state are pruned out. As a result, we may lose competing word hypotheses that have high posterior probabilities. In fact, the beam pruning method has been reported to reduce the benefit of MMIE training [12]. For our current purposes we will consider it as a baseline result.

4. POSTERIOR PROBABILITY PRUNING

In MMIE training, the Gaussian occupation count from each word hypothesis is weighted by the word posterior probability. Word hypotheses with low posterior probabilities do not contribute much to the statistics accumulation, so pruning them out should not affect the MMIE performance. Also in Sphinx lattices, there are many duplicate word hypotheses only different at entry or exit time. During MMIE training on such lattices, most of time a model just competes with itself but not other confusable models. By removing unnecessary duplicate hypotheses, we may get a more effective set of competing hypotheses for MMIE training. As shown in Figure 1, nodes and arcs are two basic elements of a word lattice; we can thus perform lattice pruning on those two levels respectively.

4.1. Arc Pruning

Each node in the lattice has a number of outgoing arcs that are transitions from the current node to succeeding nodes. We can find an arc that has the highest posterior probability within all those transitions. Then arcs (transitions) whose posterior probability is lower than a beam width multiplied by the best one will be pruned. This can be done in a forward pass from the beginning to the end of the lattice. Similarly, each node also has many incoming arcs which are transitions from preceding nodes to the current node. We can therefore do a backward pass of pruning on those incoming arcs. The posterior probability of each arc is calculated in the same way as the lattice-based forward-backward computation we described in Section 3.

4.2. Node Pruning

In addition to the posterior probability arc pruning, we can also apply the posterior probability pruning on the node level to directly remove duplicate word hypotheses in Sphinx lattices. First, the posterior probability of each node is calculated as the sum of the posterior probability of all its outgoing arcs. Then from the beginning to the end, the search of the duplicate nodes is done in the range of a 20-frame window, which is actually +/-10 frames around a node. Among these duplicate nodes, the node whose posterior probability is lower than the beam width multiplied by the best one will be pruned.

Clearly, these two levels of posterior probability pruning can be combined together to get the best result. To do this, we fixed the beam width for the node pruning and then tried different beam widths for the arc pruning.

5. EXPERIMENT SETUP

To investigate the effect of above lattice pruning methods on MMIE training, we used the WSJ-SI84 and WSJ-SI284 data set. Some information about WSJ corpus statistics is given in Table 1. The input data was a 13-order mel-scale cepstral vector (MFCC) including c_0, and then the delta and delta-delta coefficients were calculated for training. The testing data was the Nov. '92 5k-word task evaluation set. The decoder used the Lincoln Labs 5k-word closed vocabulary trigram language model from WSJ0.

Table 1. Corpus statistics for WSJ-SI84 and WSJ-SI284

Corpus	Speakers	Total Speech	Tied States	Gaussian Mixtures
WSJ-SI84	84	15h	2931	8
WSJ-SI284	284	81h	5132	16

The numerator lattices were generated by forced alignment between the correct transcription and the corresponding HMM network. The denominator lattices were generated by decoding the training utterances using a 64k-word vocabulary unigram language model trained from the WSJ0 language model data. The same unigram language model was also used for lattice probability computation. The numerator lattice was always incorporated into the denominator lattice to ensure that the denominator lattice contains the correct transcription of the utterance. In this paper, the lattices were only generated once and then used in each iteration of the EBW computation.

6. EXPERIMENT RESULTS

6.1. Baseline MMIE Results

Table 2 shows the recognition results for MMIE training compared to the initial MLE results. We can find that by applying MMIE training, the word error rate (WER) of the WSJ-SI84 system and the WSJ-SI284 system was reduced by 6.2% and 4.9%, respectively. Note that this is modest in comparison with some other reported results.

6.2. Lattice Pruning Results

The lattice pruning was only applied to the denominator lattices before they were merged with the numerator lattices. So the pruned denominator lattice still contains the correct transcription of the utterance. To perform beam pruning, different beam widths, such as 1E-80, 1E-70, 1E-60, etc. were used during the decoding of the training utterances. The lattices generated from the beam width of 1E-80 were selected as the baseline lattices to be further pruned using the posterior probability lattice pruning method. Beam width of 1E-10 was fixed for the node pruning and beam widths of 1E-70, 1E-50, etc, were tried for the arc pruning.

Table 2. Baseline MLE and MMIE results

Training Criterion	Word Error Rate (WER) %	
	WSJ-SI84	**WSJ-SI284**
MLE	6.63	4.52
MMIE	6.22	4.30

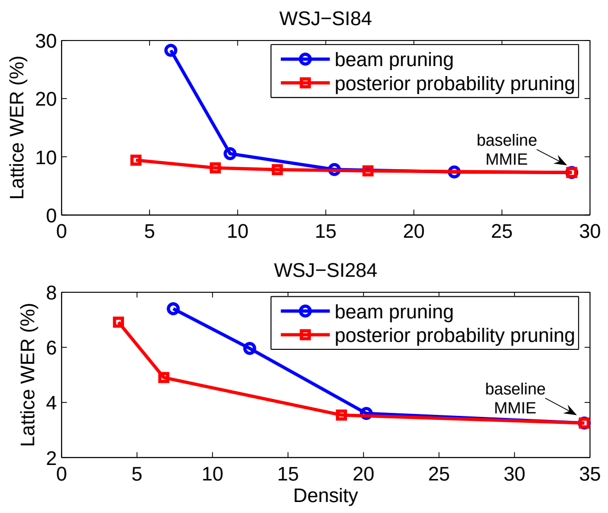

Fig. 2. The relationship between the lattice density and the lattice WER

We used the following measurements to evaluate the effect of lattice pruning:

- Lattice density: the number of nodes in the lattice divided by the actual number of words in the correct transcription. It is an estimate of the lattice complexity.
- Lattice WER: the WER of the best path in the lattice. It is a lower bound on WER which can be obtained by rescoring the lattice. We can consider it as a rough measurement of the lattice quality.
- The standard WER: the WER of the recognition results on the test set after MMIE training on pruned lattices.

Figure 2 shows the relationship between the lattice density and the lattice WER of different lattices generated from the beam pruning results and the posterior probability pruning results. It can be seen that, as the lattices become smaller, there is a very large gap between the beam pruning curve and the posterior probability pruning curve. This indicates that posterior probability pruning can produce smaller lattices of high quality, in contrast beam pruning does not.

The relationship between the lattice density and the performance of MMIE training on different lattices is shown in Figure 3. We can find that different from the beam pruning results, the performance of MMIE training with posterior probability lattice pruning is even

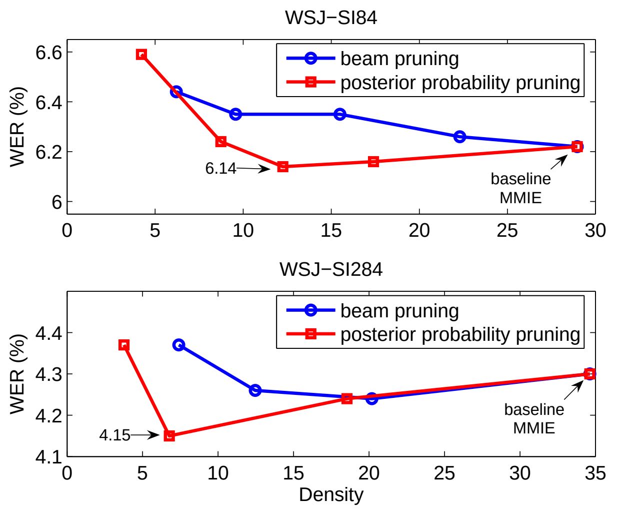

Fig. 3. The relationship between the lattice density and the MMIE performance on the Nov. '92 5k-word task evaluation set

better than the MMIE baseline results. As a result, we got a total 7.4% and 8.2% relative improvement, which is noticeably better than the numbers we reported in Section 6.1. This is because in a Sphinx lattice, there are many word hypotheses with the same word, but different entry or exit times. When doing MMIE training on such lattices, most of time a model just competes with itself but not other confusable models. By applying posterior probability pruning, we actually remove those duplicate word hypotheses from the lattice. Therefore, MMIE training can penalize the confusable models more and yield a better model.

6.3. Computation Analysis

Another advantage of MMIE training on pruned lattices is that we can save significant computation over the baseline MMIE training. In our experiment, the computation of MMIE training mainly comes from three parts—the lattice generation, the lattice format conversion and the EBW computation. Normally, the lattice generation and the lattice format conversion will dominate the overall computation. In fact, the latter one even costs more than lattice generation.

During lattice generation, the smaller the beam width, the less computation is needed. So if beam pruning is applied, then we can save some computation from this step. But this is not the case in posterior probability pruning, as we actually prune the lattice after it has been generated. However, by applying posterior probability pruning, we can save a lot of computation from the lattice format conversion and the EBW computation on the lattice. The computation of these two steps is actually proportional to the number of word hypotheses in the lattice. For example, if we pruned the lattice to half of its original size, then we can save about 50% of the running time for the lattice format conversion and the parameter update using EBW. Compared to the computation of the lattice format conversion, the computation used to prune the lattice in posterior probability pruning is very small. So generally speaking, in our experiments, when performing MMIE training with posterior probability lattice pruning, we used only 40% to 60% of the baseline MMIE running time.

7. CONCLUSION

The implementation details of MMIE training in SphinxTrain and the baseline results on various WSJ data sets were described. In addition, given that having many duplicate word hypotheses in lattices is believed to degrade the MMIE performance,we investigated MMIE training using pruned lattices. Because of the special requirements for lattices used in MMIE training: 1) unigram language model probabilities are used in training, 2) the time alignment information needs to be kept for the EBW computation, 3) the Gaussian occupation count is weighted by the posterior probability of a word hypothesis, many common lattice pruning techniques can't be applied. We proposed a posterior probability lattice pruning method to directly remove duplicate and unnecessary arcs and nodes with low posterior probabilities in the lattice. From our experiments, we found with the benefit of the posterior probability lattice pruning, MMIE training can yield more improvement, meanwhile requiring less computation than the baseline MMIE results.

8. ACKNOWLEDGMENT

This work was supported by the National Science Foundation (Grant 0713441).

9. REFERENCES

[1] "The CMU Sphinx group open source speech recognition Engines," http://cmusphinx.sourceforge.net/html/cmusphinx.php.

[2] L. R. Bahl and P. F. Brown and P. C. de Souza and R.L. Mercer, "Maximum mutual information estimation of hidden Markov model parameter for speech recognition," *Proc. ICASSP-1986*, pp. 49-52, 1986.

[3] V. Valtchev and J. J. Odell and P. C. Woodland and S. J. Young, "MMIE training of large vocabulary speech recognition system," *Speech Communication*, vol. 22, pp. 303-314, 1997.

[4] S. Ortmanns and H. Ney, "A word graph algorithms for large vocabulary continuous speech recognition," *Computer Speech and Language*, vol. 11, pp. 43-72, 1997.

[5] P. S. Gopalakrishan and D. Kanevsky and A. Nadas and D. Nahamoo, "An inequality for rational functions with applications to some statistical estimation problems," *IEEE Transactions on Information Theory*, vol. 37, pp. 107-113, 1991.

[6] Y. Liu and M. P. Harper and M. T. Johnson and and L. H. Jamieson, "The effect of pruning and compression on graphical representation of the output of a speech recognizer," *Computer Speech and Language*, vol. 17, pp. 329-356, 2003.

[7] M. Mohri and M. Riley, "Weighted determinization and minimization for large vocabulary speech recognition," *Proc. EUROSPEECH-1997*, pp. 131-134, 1997.

[8] L. Mangu and E. Brill and A. Stolcke, "Finding consensus in speech recognition: word error minimization and other applications of confusion networks," *Computer Speech and Language*, vol. 14(4), pp. 373-400, 2000.

[9] R. Schluter and R. Muller and F. Wessel and H. Ney, "Interdependence of language model and discriminative training," *Proc. IEEE ASRU Workshop*, pp. 119-122, 1999.

[10] R. Schluter and W. Macherey and B. Muller and H. Ney, "Comparison of discriminative training criteria and optimization methods for speech recognition," *Speech Communication*, vol. 34, pp. 287-310, 2001.

[11] W. Macherey and L. Haferkamp and R. Schluter and H. Ney, "Investigations on Error Minimizing Training Criteria for Discriminative Training in Automatic Speech Recognition," *INTERSPEECH-2005*, pp. 2133-2136, 2005.

[12] D. Povey, "Discriminative training for large vocabulary speech recognition," Ph.D. thesis, Cambridge University, 2004.