String and Lattice based Discriminative Training for the Corpus of Spontaneous Japanese Lecture Transcription Task

Erik McDermott & Atsushi Nakamura

NTT Communication Science Laboratories,
NTT Corporation, Kyoto-fu 619-0237, Japan
{mcd,ats}@cslab.kecl.ntt.co.jp

10.21437/Interspeech.2007-563

Abstract

This article aims to provide a comprehensive set of acoustic model discriminative training results for the Corpus of Spontaneous Japanese (CSJ) lecture speech transcription task. Discriminative training was carried out for this task using a 100,000 word trigram for several acoustic model topologies, using both diagonal and full covariance models, and using both string-based and lattice-based training paradigms. We describe our implementation of the proposal by Macherey et al. for numerical subtraction of the reference lattice statistics from the competitor lattice statistics during lattice-based Minimum Classification Error (MCE) training. We also present results for lattice-based training that does not use such subtraction, corresponding to the well-known Maximum Mutual Information (MMI) approach. Discriminative training yielded relative reductions in Word Error Rate of up to 13%. Specific problems encountered in implementing discriminative training for this task are discussed.

1. Introduction

In recent years, discriminative training of hidden Markov models (HMMs) has become a standard part of state of the art speech recognition systems. Common approaches to discriminative training include Minimum Phone Error (MPE), Maximum Mutual Information (MMI), and Minimum Classification Error (MCE) [1][2][3]. Previous work by the authors has shown that a straightforward, N-best string based implementation of MCE – with reference and competitor string sets modeled using Weighted Finite State Transducers (WFSTs) [4], as illustrated in Fig. 1 – can yield significant improvements on large-scale recognition tasks [3]. This work did not compare the N-best approach with the lattice-based training approach.

In this article we aim to provide a comprehensive set of results for discriminative training applied to the task of automatically transcribing lectures from the Corpus of Spontaneous Japanese (CSJ) database, a large vocabulary continuous speech database that has recently been used by several research groups in linguistics and speech technology in Japan [5] [6]. Here we first consider the N-best string-based discriminative training framework described in detail in [3]. We present the standard comparison of Maximum Likelihood (ML) baseline vs. discriminatively trained HMMs, for different model topologies. Results for discriminative training of full covariance HMMs are also presented. We then move to a lattice-based training approach corresponding to an extension of the framework of [3] along the lines proposed by [2]. We describe specific problems encountered in implementing discriminative training for the CSJ task.

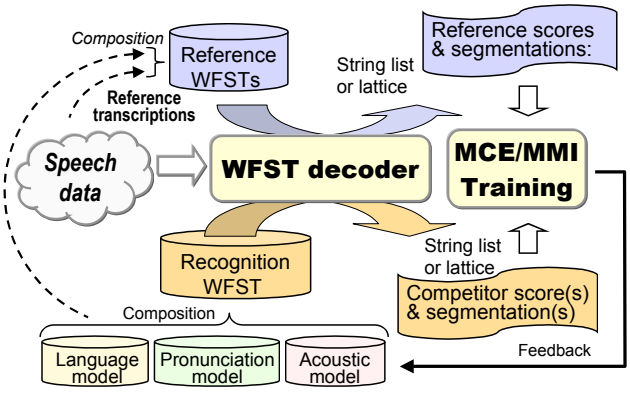

Figure 1: *Discriminative training using Weighted Finite State Transducers.*

2. String and lattice based MCE training

The overall training scheme adopted here is essentially that described in [3] and illustrated in Fig. 1.

2.1. Overall loss function

The overall loss function assumed is

$$\mathcal{F}(\mathbf{\Lambda}, \mathcal{X}) = \sum_r f(-g_{\mathcal{R}}(\mathcal{X}_r, \mathbf{\Lambda}) + g_{\mathcal{C}}(\mathcal{X}_r, \mathbf{\Lambda})), \qquad (1)$$

where $f(\cdot)$ is typically either a 0-1 sigmoid or simply a linear function [3], and where the argument to the function is a comparison of the discriminant function for the reference string set[1], $\mathcal{R}$, with that for the set of incorrect or "competitor" strings[2], $\mathcal{C}$. Here, $\mathcal{X}_r$ represents an utterance token from an overall body of training data $\mathcal{X}$, and $\mathbf{\Lambda}$ denotes the entire set of HMM parameters, including means, variances and mixing weights. Reference and competitor string sets are modeled using WFSTs (see Fig. 1). Note that if a linear loss function is used in Eq. (1), and the reference string is *not* excluded from the competitor set $\mathcal{C}$ the loss function becomes equivalent to the MMI criterion.

[1]The set of strings, including possible pronunciation variants, all considered to be correct utterance transcriptions.

[2]Typically the set of all strings allowed by the recognition grammar, minus the set of reference strings.

2.2. Definition of discriminant functions

The discriminant function given a string set $\mathcal{J}$ has the form

$$g_{\mathcal{J}}(\mathcal{X}_r, \mathbf{\Lambda}) = \frac{1}{\psi} \log \left[\sum_{j|S_j \in \mathcal{J}} e^{g_j(\mathcal{X}_r, \mathbf{\Lambda})\psi} \right]. \quad (2)$$

This definition can be used for both reference and competitor string sets, $\mathcal{R}$ and $\mathcal{C}$ respectively, and plugged into Eq. (1). Small values of the smoothing factor ψ can be used to "unweight" the top competitor strings (in a manner similar to "acoustic scaling" used in MMI studies [7]), which may help generalization compared to the 1-best string approach. The 1-best approach can be seen as the limiting case of Eq. (2), where a large value for ψ is assumed. The individual string-specific "mini"-discriminant functions $g_j(\cdot)$ are defined as

$$g_j(\mathcal{X}_r, \mathbf{\Lambda}) = \eta \log P(S_j) + \log p_{\mathbf{\Lambda}}(\mathcal{X}_r|S_j), \quad (3)$$

where $P(S_j)$ is provided by the Language Model (LM), η is the LM scaling factor, and $\log p_{\mathbf{\Lambda}}(\mathcal{X}_r|S_j)$ is defined in terms of an HMM-based log probability, summed over the Viterbi path for string S_j.

2.3. Modified Forward-Backward algorithm for lattice-based calculation of derivatives

The MCE derivatives for the 1-best and N-best versions of these definitions are straightforward [3]. Using lattices for the general case is more involved. A lattice is a compact graph representation of a set of recognition results. The Forward-Backward algorithm provides an efficient way to compute the derivative of Eq. (2) over a lattice. Space limitations prevent a full exposition, but the reader may be helped by the observations that 1) the Forward-Backward algorithm is typically used to calculate posterior probabilities efficiently, 2) the derivative of Eq. (2) in fact has the form of a posterior probability. Using Eq. (3), the derivative of Eq. (2) with respect to an individual string S_j's contribution is

$$\frac{\partial g_{\mathcal{J}}}{\partial g_j} = \frac{P(S_j)^{\eta\psi} p_{\mathbf{\Lambda}}(\mathcal{X}_r|S_j)^{\psi}}{\sum_{S_i \in \mathcal{J}} P(S_i)^{\eta\psi} p_{\mathbf{\Lambda}}(\mathcal{X}_r|S_i)^{\psi}}. \quad (4)$$

This derivative must in principle be calculated for all strings within the set $\mathcal{J}$. For the reference string set $\mathcal{R}$, corresponding arc occupancies can be calculated with the Forward-Backward algorithm so as to achieve this summation implicitly. The idea is that Eq. (4) can be broken down into lattice arc or node components whose occupancies are shared by many different strings, so as to avoid an explicit sum over strings.

The difficulty for the competitor string set $\mathcal{C}$ is that the recognition lattice often contains the reference word sequence. For a strictly correct implementation of MCE, the reference word sequence must be removed. Doing this via a physical modification of the lattice would be laborious; the numerical approach proposed in [2] is more attractive. In the interest of brevity, we refer the reader to Eq. (2) of [2], describing the word arc occupancy corresponding to the lattice-factored MCE derivative.

In outline, the numerical lattice subtraction method consists in 1) calculating the forward-backward scores on the entire recognition lattice, which is assumed to contain the reference string; 2) separately calculating forward-backward scores for a lattice containing only alignments of the reference string; 3) subtracting the total forward-backward score obtained in (2) from the total forward-backward score obtained in (1) and re-normalizing all arc occupancies obtained in both (1) and (2) by the new score; 4) subtracting at the state level all re-normalized statistics in (2) from the re-normalized statistics in (1). Note that while the subtraction in (3) concerns two scalars, the subtraction in (4) is done for all states found in the lattices. Furthermore, the subtraction in (4) does not require that lattice arcs or states be paired up; the relevant statistics are first added for every state in (1), and later subtracted for every state in (2). Finally, note that here we have only described the modified Forward-Backward calculation of the derivative for the competitor string set $\mathcal{C}$. The (unmodified) Forward-Backward algorithm must also be used separately to compute the derivatives for the reference set $\mathcal{R}$.

A crucial step is the determination of the part of the recognition lattice that corresponds to the reference word sequence. For this purpose we employed WFST composition, which generated "subtraction" lattices that we then used as just described.

2.4. Optimization using gradient descent

In contrast with a large body of work that uses the Extended Baum Welch (EBW) algorithm for discriminative training [2][7], all results presented here use Rprop, a simple second-order gradient-based optimization method, for which good results have been reported [3][8][9]. (Among several studies comparing Rprop with EBW, it was found in [9] that Rprop outperformed EBW).

3. Corpus of Spontaneous Japanese

The Corpus of Spontaneous Japanese (CSJ) lecture speech transcription task is a large-scale, spontaneous speech recognition task based on lectures recorded from presentations at scientific conferences in Japan [6].

3.1. Speech data

The standard CSJ male+female *A* set (about 190,000 utterances, approximately 230 hours of audio data) from 967 lectures was used for training. The test set consists of 10 lecture speeches, each from a different speaker, comprising 130 minutes of audio in total. The feature vectors used consisted of 39 MFCC, delta and delta-delta components, computed using lecture-based cepstral mean subtraction.

3.2. Text data & word segmentation

The LMs typically used on this task are made from text data corresponding to 2672 lectures within the CSJ database. Transcriptions exist for each lecture, including kanji/kana transcription, phonetic transcription, and part of speech annotation for each word. These three components are concatenated for each word entry; the LM is therefore sensitive to pronunciation and part of speech.

An important issue in handling Japanese text is word segmentation. This issue directly affects discriminative training. The point is that the reference transcriptions given to the training procedure as the target string must in principle correspond to a possible word sequence in the recognition LM.

The CSJ database comes with a standard word segmentation. However, these segmentations only exist at the level of entire lectures. For each utterance (within a lecture), a different, phrase-based segmentation is used. In our work, in order to generate references (and reference WFSTs) for each utterance, the phrase-segmented kanji/kana transcription for each utter-

ance was aligned to the corresponding utterance segment in the word-segmented lecture transcription, resulting in a word-level segmentation for each utterance that matches the units used in the design of the LM, and that therefore can be used as a reference for discriminative training. Note that this time-consuming issue does not affect ML training, as the latter simply uses the phonetic transcription for each utterance.

4. Language model during training & testing

The test data results described in the following are all based on a 100k word trigram using original Kneser-Ney back-off smoothing, and with a perplexity of 75.3. The out of vocabulary rate on the test set is 1.36%. The LM scaling weight during testing was 13, and the beam width was 200. This corresponds to decoding at around 5 times real time.

Previous work [3] used a 68,000 word unigram during MCE training on this task; it was later found that using the target 100,000 trigram, described above, consistently yielded better results. Hence, in this study, the LM used during MCE training is the 100,000 word trigram described above. The trigram LM was represented as two component WFSTs and used in the fast on-the-fly composition one-pass decoding algorithm described in [10].

5. Discriminative optimization of acoustic models

For all the training configurations examined, batch-mode Rprop was used to optimize the HMM parameters using the gradient of the MCE cost function in the scheme described in [3] (see Fig. 1). At most 5 iterations of MCE training were carried out, and the resulting HMMs tested. (Preliminary tests of successive models made after each iteration indicated a performance plateau after 5 iterations, with performance degrading after 9-10 iterations). Both diagonal and full-covariance triphone HMMs were used, with model topologies of 2000 to 5000 states, and numbers of Gaussians per state ranging from 8 to 32.

5.1. String-based MCE training

5.1.1. Diagonal covariance models

The LM scaling factor during training was 13 (the same as during testing); the beam size during training was 140 (compared to 200 during testing). With these settings, the recognition pass during training can be performed at around 0.4 times real time. Only the single top incorrect string candidate was used for the computation of the MCE gradient. A linear loss function was used (non-linear loss functions were previously investigated on CSJ, but not found to help, in contrast to results for other tasks [3]). The results for both the MCE-trained models (after 5 training iterations, requiring less than 20 hours using 25-30 processors) and the Maximum Likelihood (ML) baselines are shown in Table 1.

5.1.2. Full covariance models

On the CSJ task, ML-trained full covariance acoustic models yield significant gains over diagonal covariance models. In our implementation, the covariance matrices are Gaussian-specific, i.e. we do not use tying of any kind.

We investigated the effectiveness of discriminative training applied to full covariance models [11]. The MCE gradient for full covariance models is a straightforward extension of the MCE gradient for diagonal covariance models, detailed in [3]. Given an inverse Cholesky decomposition of the full covariance matrix, $\mathbf{W}_{j,m}^{-1} = \mathbf{L}_{j,m}\mathbf{L}_{j,m}^{'}$, (for a Gaussian m within a state j), previous work detailing the matrix calculus involved can be used [12]:

$$\frac{\partial b_{j,m}(\mathbf{x}_{k,t})}{\partial \mathbf{L}_{j,m}} = \quad (5)$$
$$b_{j,m}(\mathbf{x}_{k,t}) \left\{ (\mathbf{L}_{j,m}^{-1})' - (\mathbf{x}_{k,t} - \boldsymbol{\mu}_{j,m})(\mathbf{x}_{k,t} - \boldsymbol{\mu}_{j,m})' \mathbf{L}_{j,m} \right\},$$

with only the diagonal elements of $(\mathbf{L}_{j,m}^{-1})'$ necessary for implementation. Here $\mathbf{x}_{k,t}$ refers to a feature vector obtained at time t, for utterance token $\mathcal{X}_r$, $b_{j,m}(\cdot)$ refers to a Gaussian pdf, and $\boldsymbol{\mu}_{j,m}$ refer to the mean vector for this Gaussian. This expression is easily plugged in to the calculation of the overall MCE gradient [3].

MCE training was carried out for full covariance models using a decoding beam of 100. As for diagonal covariance training, the LM weight is 13, only the single incorrect string candidate was used for the computation of the MCE gradient, and a linear loss function was used. With these settings, decoding can be performed at around real-time. The results for both the MCE-trained models (after 5 training iterations, requiring up to 3 days using 25-30 processors), and the Maximum Likelihood (ML) baselines, are shown in Table 2.

Table 1: Error rates for string-based MCE vs. ML baseline and (in parentheses) Relative Error Rate Reductions (%) compared to ML performance. Diagonal covariance models were used.

# States	# Gaussians/state	ML	MCE
2000	16	23.0	20.3 (11.7)
3000	16	22.4	20.4 (8.9)
4000	16	22.1	19.9 (10.0)
5000	32	21.6	19.6 (9.3)

Table 2: Error rates for string-based MCE vs. ML baseline and (in parentheses) Relative Error Rate Reductions (%) compared to ML performance. Full covariance models were used.

# States	# Gaussians/state	ML	MCE
2000	8	20.2	19.4 (4.0)
2000	16	19.8	18.9 (4.6)
3000	8	20.1	19.0 (5.5)
3000	16	19.4	18.9 (2.6)
3000	32	20.7	-

5.2. Lattice-based MCE training

The lattice-based MCE training procedure described in Sec. 2.3 was investigated.

5.2.1. Diagonal covariance models

Lattices of on average 150 arcs/frame were generated once for the entire training data (beam size during lattice generation was set at 210). A number of smoothing factors ψ were evaluated in the $[0.02, 1.0]$ range. For each choice of ψ, four iterations of Rprop were run for both the MCE-oriented lattice subtraction method outlined earlier, and lattice-based discriminative training without the subtraction, corresponding to MMI-

based optimization. Depending on the value of ψ, we confirmed that lattice-based MCE training behaved differently from MMI-based training; MCE tended to yield better performance for the higher values of ψ, but for the lower values, the differences were small and showed no clear trend either way. Choosing ψ to optimize each method yielded identical performances (the optimal value for each method was $\psi = 0.03$). As for the string-based training described earlier, the loss function was set to be linear for both diagonal and full covariance lattice-based training. Only the 5000 state, 32 Gaussians/state model previously used for string-based training was used. The results are shown in Table 3.

5.2.2. Full covariance models

Lattices of on average 110 arcs/frame were generated for the entire training data (beam size during lattice generation was set at 210). In order to speed up computation, both arc pruning and gaussian pruning based on occupancy were used. Three iterations of Rprop[3] were carried out for both the lattice-based subtraction method (MCE) and discriminative training without the subtraction method (MMI). The lattice smoothing factor was set at $\psi = 0.04$. Only the 2000 state, 16 Gaussians/state model previously used for string-based training was used. The results are shown in Table 3.

Table 3: Error rates for lattice-based MCE and MMI vs. ML baseline and (in parentheses) Relative Error Rate Reductions (%) compared to ML performance.

Model	ML	MCE	MMI
5000x32-diag	21.6	18.7 (13.4)	18.7 (13.4)
2000x16-full	19.8	18.5 (6.6)	18.6 (6.1)

6. Summary

A number of different configurations were examined for discriminative training on the Corpus of Spontaneous Japanese (CSJ) lecture transcription task, including different acoustic model topologies, full vs. diagonal covariance models, and string- vs. lattice- based training paradigms. Evaluated with a 100k word trigram LM on this task, MCE training yielded up to 13% relative reductions in error. Discriminative training of full-covariance models yielded smaller relative improvements compared to the ML baseline than for the diagonal covariance case, but produced the best overall word accuracies of 81.1% and 81.5% for the string-based and lattice-based paradigms, respectively. Lattice-based training yielded significant improvements over string-based training for both diagonal and full covariance models. Note that lattice-based training proceeds quite a bit faster than string-based training, as the lattices are only generated once and then recycled during training. Only one acoustic model topology and only one setting for the lattice smoothing were considered for the lattice-based full covariance experiments; investigation of different settings may yield further improvements in performance. It is interesting to see that even though the lattice-based approach yielded the best results, the string-based approach using just a single competitor string consistently yielded solid improvements as well. Comparison of the lattice subtraction method proposed by Macherey et al. with lattice-based training without subtraction (corresponding to a comparison between lattice-based MCE and MMI) has not revealed important differences on this task; nevertheless the overall trend we observed (reflected in the results for full-covariance lattice-based training) was that lattice subtraction yielded small improvements. Note that we have labeled the approach that does not use lattice subtraction as "MMI", but this could equally be termed "approximated MCE", in contrast to "exact MCE", for which the lattice subtraction is performed rigorously. We believe that these results can serve as a useful reference for other researchers aiming to implement discriminative training for this task.

[3]The reader may be interested to learn that even a single iteration of Rprop run in semi-batch mode yielded significant improvements.

7. References

[1] D. Povey, *Discriminative Training for Large Vocabulary Speech Recognition*, Ph.D. thesis, University of Cambridge, 2004.

[2] W. Macherey, L. Haferkamp, R. Schlueter, and H. Ney, "Investigations on Error Minimizing Training Criteria for Discriminative Training in Automatic Speech Recognition," in *Proc. Eurospeech*, 2005.

[3] E. McDermott, T.J. Hazen, J. Le Roux, A. Nakamura, and S. Katagiri, "Discriminative training for large vocabulary speech recognition using Minimum Classification Error," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 15, no. 1, pp. 203–223, January 2007.

[4] M. Mohri, F. Pereira, and M. Riley, "Weighted Finite State Transducers in Speech Recognition," in *Proc. of Automatic Speech Recognition Workshop*, 2000, pp. 97–106.

[5] T. Kawahara, H. Nanjo, T. Shinozaki, and S. Furui, "Benchmark test for speech recognition using the corpus of spontaneous japanese," in *Proc. of the Spontaneous Speech Processing & Recognition Workshop*, Tokyo, 2003, pp. 135–138.

[6] K. Maekawa, H. Koiso, S. Furui, and H. Isahara, "Spontaneous Speech Corpus of Japanese," in *Proc. Second International Conference on Language Resources and Evaluation (LREC)*, 2000, pp. 947–952.

[7] P.C. Woodland and D. Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech and Language*, vol. 16, pp. 25–47, 2002.

[8] J. Droppo and A. Acero, "Joint Discriminative Front End and Back End Training for Improved Speech Recognition Accuracy," in *Proc. IEEE ICASSP*, 2006, vol. 1, pp. 281–284.

[9] R. Teunen, *Acoustic Modeling for Automatic Speech Recognition: Deriving Discriminative Gaussian Networks*, Ph.D. thesis, Stanford University, Department of Electrical Engineering, 2002.

[10] T. Hori and A. Nakamura, "Generalized fast on-the-fly composition algorithm for WFST-based speech recognition," in *Proc. Eurospeech*, 2005, pp. 557–650.

[11] D. Povey, "SPAM and full covariance for speech recognition," in *Proc. Interspeech*, 2006, pp. 1159–1163.

[12] V. Valtchev, *Discriminative Methods in HMM-based Speech Recognition*, Ph.D. thesis, University of Cambridge, 1995.