ACOUSTICALLY DISCRIMINATIVE TRAINING FOR LANGUAGE MODELS

Gakuto KURATA, Nobuyasu ITOH and Masafumi NISHIMURA

IBM Research, Tokyo Research Laboratory, IBM Japan, Ltd.
1623-14 Shimotsuruma Yamato-shi Kanagawa, 242-8502, Japan

{gakuto,iton,nisimura}@jp.ibm.com

ABSTRACT

This paper introduces a discriminative training for language models (LMs) by leveraging phoneme similarities estimated from an acoustic model. To train an LM discriminatively, we needed the correct word sequences and the recognized results that Automatic Speech Recognition (ASR) produced by processing the utterances of those correct word sequences. But, sufficient utterances are not always available. We propose to generate the probable N-best lists, which the ASR may produce, directly from the correct word sequences by leveraging the phoneme similarities. We call this process the "**Pseudo-ASR**". We train the LM discriminatively by comparing the correct word sequences and the corresponding N-best lists from the Pseudo-ASR. Experiments with real-life data from a Japanese call center showed that the LM trained with the proposed method improved the accuracy of the ASR.

Index Terms— Discriminative Training, Language Model, Phoneme Similarity, Finite State Transducer

1. INTRODUCTION

In order to reduce the error of the Automatic Speech Recognition (ASR), discriminative training is widely used for acoustic models (AMs) [1, 2], language models (LMs) [3, 4, 5, 6], and decoding graphs [7, 8]. To conduct discriminative training, we needed the correct word sequences and corresponding utterances. In discriminative training, we had the ASR recognize the utterances and got the results. Then by comparing the correct word sequences and the recognized results, we discriminatively trained the AM, the LM, or the decoding graph.

Considering that building an AM is expensive and time-consuming, once we build an AM for a certain environment, such as telephony environment, we sometimes use it for several applications. As regards an LM, we need to build new ones for each application. It's obvious that Business-Finder [9] and Help-Desk [10] cannot be implemented with the same LM even though both are telephony applications[1]. In this paper, we consider the situation that we are building a new application by building an application-specific LM and reusing a general AM. In this case, we can only use the text data that was collected to build an LM. Sufficient utterances to build an AM or to discriminatively train an AM, an LM or a decoding graph are not available.

We propose a new framework for discriminative training of an LM, which doesn't require utterances. The key idea of this paper is to obtain probable N-best lists directly from the correct word sequences by leveraging phoneme similarities estimated from the AM and to discriminatively train the LM based on these N-best lists. We call the former process the "**Pseudo-ASR**". In the Pseudo-ASR, because we estimate the phoneme similarities from the AM which are used to recognize the test utterances (in real deployment), we can expect to obtain "reliable" erroneous N-best lists.

[1] An application-specific AM is preferable, but a general one can work.

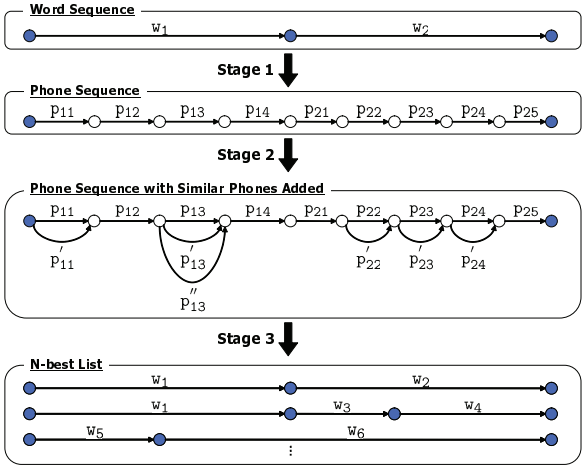

Figure 1. Intuitive Flow of Pseudo-ASR

2. PROPOSED METHOD

In this section, we describe our proposed method, in which no utterance data is required to train the LM discriminatively. In the proposed method, we have the Pseudo-ASR generate the N-best lists from the correct word sequences. Then we train the LM discriminatively by comparing the correct word sequences and the corresponding N-best lists. In the rest of this section, we explain the Pseudo-ASR in detail and the discriminative training of the LM based on the N-best lists from the Pseudo-ASR.

2.1. Pseudo-ASR

Fig. 1 shows the intuitive flow of the Pseudo-ASR:

Stage 1. A word sequence "$w_1 w_2$" is converted into a phone sequence "$p_{11} \cdots p_{14} p_{21} \cdots p_{25}$" by consulting the lexicon[2].

Stage 2. Similar phones whose similarities are estimated from the AM are added with probabilities to the phone sequence[3]. For example, the phone "p'_{13}" and "p''_{13}" which are similar to "p_{13}" are added.

Stage 3. Combined with the lexicon which converts a phone sequence into a word and the LM which assigns a probability to a word sequence, an N-best list of word sequences can be produced from the phone sequence.

In order to compare a standard ASR and the Pseudo-ASR, we briefly explain how the standard ASR produces the N-best list. Given the speech signal $\boldsymbol{X}$, the standard ASR produces the h-th best hypothesized word sequence $\boldsymbol{W_h}$ that satisfies the following Equation (1):

[2] The lexicon contains the pairs of a word and its corresponding phone sequence

[3] Probability values are not depicted in **Fig. 1** to avoid confusion.

$$\boldsymbol{W_h} = \underset{\boldsymbol{W} \neq \boldsymbol{W_1}, \cdots, \boldsymbol{W_{h-1}}}{\operatorname{argmax}} g(\boldsymbol{X}, \boldsymbol{W_h}; \Lambda, \Gamma) , \quad (1)$$

$$\text{where} \quad g(\boldsymbol{X}, \boldsymbol{W_h}; \Lambda, \Gamma) = \alpha \log P(\boldsymbol{X}|\boldsymbol{W_h}, \Lambda) + \log P(\boldsymbol{W_h}|\Gamma) . \quad (2)$$

Λ is the AM, Γ is the LM, and α is the inverse of the LM weight. In the proposed Pseudo-ASR, we estimate the first term of the right side of Equation (2) based on the similarities between the phones. Considering the intuitive flow in **Fig. 1**, the first term is embedded in the process of adding the similar phones (**Stage 2**).

We explain each stage of the Pseudo-ASR from the viewpoint of implementation. In **Stage 1**, we convert a correct word sequence into a phone sequence by looking up the lexicon. We express a phone sequence as a finite state acceptor (FSA) $\mathcal{PS}$. **Stage 2** and **Stage 3** are implemented as the composition of several finite state transducers (FSTs). The following composition (3) is calculated for each input phone sequence $\mathcal{PS}$ and a Viterbi search is conducted over $\mathcal{WG}$, producing the N-best list.

$$\mathcal{WG} = (((\mathcal{PS} \circ \mathcal{PP}) \circ \mathcal{LX}) \circ \mathcal{LM}) . \quad (3)$$

The FSTs are defined as:

$\mathcal{PP}$	:	Phone to Phone FST	$\mathcal{LM}$	:	LM
$\mathcal{LX}$	:	Lexicon	$\mathcal{WG}$	:	Word Graph

Stage 2 corresponds to the composition of $(\mathcal{PS} \circ \mathcal{PP})$ and **Stage 3** to the remaining compositions and the search over the $\mathcal{WG}$.

We explain how each FST is prepared.

2.1.1. $\mathcal{PP}$: Phone to Phone FST

This FST adds phones to the $\mathcal{PS}$ based on the similarities between the phone in the $\mathcal{PS}$ and the other phones. Intuitively, similar phones are more likely to be misclassified in the ASR[11, 12]. In other words, this FST simulates the AM of the ASR in the process of Pseudo-ASR.

The similarities between each phone are estimated from the AM. In the AM, each phone is represented as a 3-state, left-to-right HMM. Each state of the HMM is modeled by a mixture of Gaussians. For each phone, we select the Gaussian with the biggest weight in the middle state as the representative Gaussian of this phone. Then for each pair of the phones, we calculate the Bhattacharrya Distance (BD) between the representative Gaussians and regard this as the distance between the phones [13]. The BD between two Gaussians $N(\boldsymbol{\mu_i}, \boldsymbol{\Sigma_i})$ and $N(\boldsymbol{\mu_j}, \boldsymbol{\Sigma_j})$is defined as:

$$\text{BD} = \frac{1}{8}\boldsymbol{\mu_{ij}}^T(\frac{\boldsymbol{\Sigma_i} + \boldsymbol{\Sigma_j}}{2})^{-1}\boldsymbol{\mu_{ij}} + \frac{1}{2}\ln\frac{|(\boldsymbol{\Sigma_i} + \boldsymbol{\Sigma_j})/2|}{|\boldsymbol{\Sigma_i}|^{\frac{1}{2}}|\boldsymbol{\Sigma_j}|^{\frac{1}{2}}} , \quad \text{where} \quad \boldsymbol{\mu_{ij}} = \boldsymbol{\mu_i} - \boldsymbol{\mu_j} .$$

The probability $prob(\mathtt{p_j}|\mathtt{p_i})$ that the phone $\mathtt{p_i}$ is replaced with $\mathtt{p_j}$ is defined based on the BD bd_{ij} between them as:

$$prob(\mathtt{p_j}|\mathtt{p_i}) = exp(-1 \times bd_{ij}) / \sum_k exp(-1 \times bd_{ik}) .$$

Since the BD between a phone and itself is 0, the probability $prob(\mathtt{p_i}|\mathtt{p_i})$, meaning that the phone $\mathtt{p_i}$ is not replaced with another phone, is the highest among $prob(\mathtt{p_k}|\mathtt{p_i})$ for all k.

Our set of phones contains the phone of silence $\mathtt{p_{SIL}}$. By leveraging this $\mathtt{p_{SIL}}$, we can handle the insertion and the deletion of the phones. The insertion of the phone $\mathtt{p_i}$ can happen at any position in $\mathcal{PS}$ with $prob(\mathtt{p_i}|\mathtt{p_{SIL}})$. The probability of the deletion of the phone $\mathtt{p_i}$ is $prob(\mathtt{p_{SIL}}|\mathtt{p_i})$.

Fig. 2 shows a subset of $\mathcal{PP}$. The $\mathcal{PP}$ is represented as a self-loop FST. The transition "$\mathtt{p_i} : \mathtt{p_j}$ / $prob(\mathtt{p_j}|\mathtt{p_i})$" means that this FST

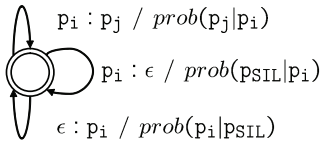

Figure 2. Subset of Phone to Phone FST $\mathcal{PP}$

accepts $\mathtt{p_i}$ and outputs $\mathtt{p_j}$ with the probability $prob(\mathtt{p_j}|\mathtt{p_i})$. The transition "$\mathtt{p_i} : \epsilon$ / $prob(\mathtt{p_{SIL}}|\mathtt{p_i})$" expresses the deletion of $\mathtt{p_i}$, meaning that this FST accepts $\mathtt{p_i}$ and outputs no phone with the probability $prob(\mathtt{p_{SIL}}|\mathtt{p_i})$. The transition "$\epsilon : \mathtt{p_i}$ / $prob(\mathtt{p_i}|\mathtt{p_{SIL}})$" expresses the insertion of $\mathtt{p_i}$, meaning that this FST accepts no phone and outputs $\mathtt{p_i}$ with the probability $prob(\mathtt{p_i}|\mathtt{p_{SIL}})$.

In order to reduce the computational cost, we limit the number of pairs of the phones included in the $\mathcal{PP}$. First, we sort all of the pairs in descending order of $prob(\mathtt{p_j}|\mathtt{p_i})$. Then we select the top C pairs.

2.1.2. $\mathcal{LX}$: Lexicon

This FST converts a phone sequence into a word. An $\mathcal{LX}$ is constructed from the lexicon. For example, this FST accepts the phone sequence "`S P IY CH`" and outputs the corresponding word "speech". After the composition of $((\mathcal{PS} \circ \mathcal{PP}) \circ \mathcal{LX})$, the input phone sequence $\mathcal{PS}$ is converted to a word sequence.

2.1.3. $\mathcal{LM}$: LM

This FST assigns probabilities to word sequences. An LM can be represented as an FST [14].

2.1.4. $\mathcal{WG}$: Word Graph

With the composition (3), a word graph $\mathcal{WG}$ is constructed. Then by performing a Viterbi search over this word graph, the N-best list for the input phone sequence $\mathcal{PS}$ is produced.

2.2. Discriminative Training

By comparing the correct word sequence and the corresponding N-best list generated by the Pseudo-ASR, we train the LM discriminatively. As an algorithm for discriminative training of the LM using speech data, the article [4] describes the algorithm based on the correct word sequence and the N-best list from the ASR. We leverage this algorithm by replacing the ASR with the Pseudo-ASR.

Given a correct word sequence $\boldsymbol{W_0}$ and its corresponding phone sequence $\boldsymbol{PS}$, the h-th best hypothesized word sequence $\boldsymbol{W_h}$ of the Pseudo-ASR satisfies the following Equation (4):

$$\boldsymbol{W_h} = \underset{\boldsymbol{W} \neq \boldsymbol{W_1}, \cdots, \boldsymbol{W_{h-1}}}{\operatorname{argmax}} g_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_h}; \Lambda, \Gamma) , \quad (4)$$

$$\text{where} \quad g_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_h}; \Lambda, \Gamma) = \alpha \log P(\boldsymbol{PS}|\boldsymbol{W_h}, \Lambda) + \log P(\boldsymbol{W_h}|\Gamma) . \quad (5)$$

Λ is the AM, Γ is the LM, and α is the inverse of the LM weight. Note that the first term of Equation (5) is estimated by multiplying the probabilities "$prob(\mathtt{p_j}|\mathtt{p_i})$" based on the similarities between the phones, which is a different approach from Equation (2).

The misclassification function is defined as follows:

$$d(\boldsymbol{PS}; \Lambda, \Gamma) = -g_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_0}; \Lambda, \Gamma) + G_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_1}, \cdots, \boldsymbol{W_N}; \Lambda, \Gamma) ,$$

where the anti-discriminant function based on the N-best list is defined as:

$$G_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_1}, \cdots, \boldsymbol{W_N}; \Lambda, \Gamma) = \log(\frac{1}{N}\sum_{r=1}^{N}\exp[g_{pseudo}(\boldsymbol{PS}, \boldsymbol{W_r}; \Lambda, \Gamma)\eta])^{\frac{1}{\eta}} .$$

Table 1. Statistics of Test Data / CER

call ID	Test Data: Gender	Test Data: # of characters	CER: Baseline (0 iteration)	CER: Proposed (25 iterations)
A	F	1,611	20.2	19.8
B	M	2,122	28.8	27.6
C	F	1,934	35.6	33.6
D	M	1,196	18.4	18.1
E	F	1,631	26.5	26.1
F	M	1,887	28.2	28.5
G	F	3,694	37.1	36.5
H	F	1,248	35.4	34.6
Total	–	15,323	30.2	29.4

η controls how the different hypotheses are weighted. A sigmoid function which limits the range from 0 to 1 is used for the class loss function:

$$l(\boldsymbol{PS}) = l(d(\boldsymbol{PS})) = \frac{1}{1+\exp(-\gamma d(\boldsymbol{PS})+\theta)} ,$$

where γ and θ are the parameters of the sigmoid function. By the generalized probabilistic descent (GPD) algorithm, the parameters of the LM can be updated iteratively with the learning rate s as:

$$\Gamma_{t+1} = \Gamma_t - s\nabla l(\boldsymbol{PS}; \Lambda_t, \Gamma_t) .$$

By focusing only on the LM while not changing the AM, the gradient of the loss function becomes:

$$\nabla l = \frac{\partial l}{\partial d}\frac{\partial d(\boldsymbol{PS}; \Lambda, \Gamma)}{\partial \Gamma} . \quad (6)$$

The first term of Equation (6) is based on the sigmoid function:

$$\frac{\partial l}{\partial d} = \gamma l(d)(1-l(d)) .$$

The second term can be calculated based on the frequency of the word n-gram sequence $\mathbf{w}$ appearing in the correct word sequence $\boldsymbol{W_0}$ and its corresponding N-best list produced by the Pseudo-ASR:

$$\frac{\partial d(\boldsymbol{PS}; \Lambda, \Gamma)}{\partial p_{\mathbf{w}}} = [-I(\boldsymbol{W_0}, \mathbf{w}) + \sum_{r=1}^{N} C_r I(\boldsymbol{W_r}, \mathbf{w})] ,$$

$$\text{where} \quad C_r = \frac{\exp[g(\boldsymbol{PS}, \boldsymbol{W_r}; \Lambda, \Gamma)\eta]}{\sum_{j=1}^{N}\exp[g(\boldsymbol{PS}, \boldsymbol{W_j}; \Lambda, \Gamma)\eta]}$$

and $I(\boldsymbol{W_k}, \mathbf{w})$ $(k = 0, 1, \cdots, N)$ denotes the frequency of the word n-gram sequence $\mathbf{w}$ appearing in the word sequence $\boldsymbol{W_k}$.

3. EXPERIMENT

We conducted an ASR experiment to verify whether the LM trained discriminatively with the proposed method improved the accuracy of the ASR. We describe our experiment here. Then we show our results and discuss them.

3.1. Experimental Setup

We conducted the experiment using real-life data from a Japanese call center. We randomly selected 8 calls and used the utterances of the agents as the test data[4]. The left side of **Table 1** shows the statistics of the test data. The first column is the call ID, the second is the gender of the agent and the third is the number of characters in the transcribed text.

We built the AM of 57 phones for the telephony environment, estimated the baseline 3-gram LM with modified Kneser-Ney smoothing [15] from the corpus of 234,998 sentences, and prepared the lexicon of 20,652 words with 22,132 pronunciations. The baseline ASR is composed of these AM, LM, and lexicon.

[4]We decided not to use the utterances of the customers because they speak so freely and variably that the accuracy of ASR tends to be extremely poor.

Table 2. Coverage Ratios of 1,2,3-grams in 100-best from Baseline ASR by Those in 100-best from Pseudo-ASR.

	1-gram	2-gram	3-gram
Coverage	0.977	0.742	0.677

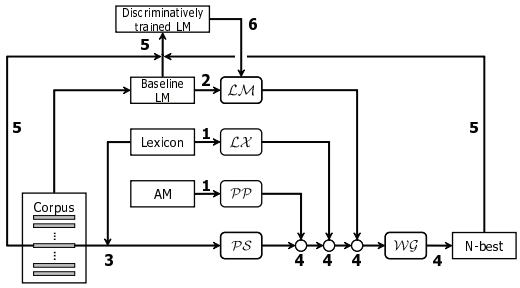

Figure 3. Flow of Experiment

3.2. Performance of Pseudo-ASR

The performance of discriminative training is affected by how close the N-best lists from the Pseudo-ASR are compared to those from the ASR. In order to examine how the Pseudo-ASR works, we conducted a preliminary experiment. We selected 275 utterances from the test utterances. We had the baseline ASR produce the 100-best lists from the utterances and had the Pseudo-ASR produce the 100-best lists from the transcriptions of the same utterances. Then we investigated how well the 1-grams, 2-grams, and 3-grams included in the 100-best lists from the baseline ASR are covered by those from the Pseudo-ASR. The coverage ratios are shown in **Table 2**. Because the Pseudo-ASR and the baseline ASR are not the same, the coverage ratios were smaller than 1. However, considering that the Pseudo-ASR didn't use any utterances to generate the N-best lists, it produced similar results to the results of the baseline ASR. Therefore we can expect that the N-best lists from the Pseudo-ASR contribute to discriminative training for LMs.

3.3. Flow of Experiment

Fig. 3 provides the flow of our experiment. The numbers in **Fig. 3** correspond to the numbers of the following steps:

Step 1. Prepared the $\mathcal{LX}$ from the lexicon. Estimated the $\mathcal{PP}$ from the AM based on the similarities between the phones as written in Section 2.1.1. Note that the same $\mathcal{LX}$ and the same $\mathcal{PP}$ were used in the following iterations.

Step 2. Converted the baseline LM to the FST $\mathcal{LM}$.

Step 3. Randomly selected 500 sentences from the corpus[5] and converted each sentence into a phone sequence by looking it up in the lexicon. Each phone sequence is expressed as an FSA $\mathcal{PS}_i (i = 1, 2, \cdots, 500)$.

Step 4. Had the Pseudo-ASR generate an N-best list from each $\mathcal{PS}_i$.

Step 5. Trained the LM discriminatively based on the selected 500 sentences and their corresponding N-best lists.

Step 6. Converted the discriminatively trained LM to the FST and returned to **Step 3**.

We trained the LM iteratively multiple times by iterating **Step 3** to **Step 6**. Note that in **Step 5**, the baseline LM was discriminatively trained in the first iteration and then the LM trained in the previous iteration was trained discriminatively in the following iteration.

After each iteration, we replaced the baseline LM in the baseline ASR with the discriminatively trained LM and evaluated the ASR accuracy with the test data.

[5]Equivalent to the "Semi-Batch" mode in the article [16]

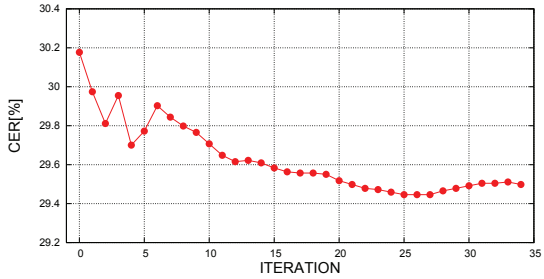

Figure 4. CER over Iterations

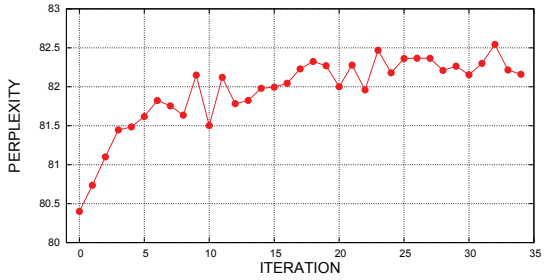

Figure 5. Perplexity over Iterations

For the constant values of the Pseudo-ASR, we set the number of the pairs of the phones in $\mathcal{PP}$ to $C = 500$ and the number of the N-best list to $N = 100$. For the parameters of discriminative training, we set $s = 0.001$, $\eta = 0.1$, $\gamma = 0.5$, and $\theta = 0$. We decided on these values empirically based on a pilot experiment.

3.4. Evaluation and Discussion

First, we explain the criterion for evaluation. To measure the ASR accuracy, we used the Character Error Ratio (CER). The reason is that ambiguity exists in word segmentation in Japanese. For example, "東京都知事 (Governor of Tokyo)" can be segmented into words in four ways: (1) "東京都知事", (2) "東京都 / 知事", (3) "東京 / 都知事", and (4) "東京 / 都 / 知事". In all cases, the same characters are used and the number of characters remains 5. However, the number of words seems to change from 1 to 3 because of the ambiguity, so the Word Error Rate (WER) fluctuates accordingly. Therefore, the CER is a more suitable criterion in Japanese.

The right side of **Table 1** on the previous page shows the CER for the test data. The fourth column is the CER of the baseline ASR. The fifth column is the CER after 25 iterations of discriminative training, which resulted in the best accuracy. **Fig. 4** shows the average CER for the test data over the iterations[6]. **Fig. 5** shows the perplexity for the transcribed test data over the iterations.

Though the proposed method doesn't require any speech data to train the LM discriminatively, there was an improvement from 30.2% to 29.4% in the CER. This improvement was statistically significant at the 5% level. The perplexity increased slightly over the iterations because the proposed discriminative training is not intended to reduce the perplexity. The increase of the perplexity didn't have any negative effect on the CER.

4. CONCLUSION

We proposed an acoustically discriminative training for an LM without using the speech data. In our proposed method, we generate the probable erroneous N-best list of word sequences directly from the correct word sequence by using the Pseudo-ASR that leverages the similarities between phones in the AM. Then we train the LM discriminatively based on the correct word sequences and the corresponding N-best lists. Note that multiple iterations of the Pseudo-ASR on many word sequences are computationally heavy, but this process is highly parallelizable.

We conducted an experiment with real-life data from a Japanese call center. The results showed that the proposed method improved the accuracy of the ASR, even though the proposed method doesn't require any speech data to train the LM discriminatively.

In the proposed method, we ignored the phone contexts when estimating the similarities between phones. Taking the phone contexts into consideration may improve the performance. Other sophisticated techniques to estimate the similarities between phones [17, 18, 19] may also help.

[6]Iteration 0 means the baseline ASR

5. REFERENCES

[1] Phil C. Woodland and Daniel Povey, "Large scale discriminative training of hidden Markov models for speech recognition," *Computer Speech & Language*, vol. 16, 2002.

[2] Daniel Povey and Brian Kingsbury, "Evaluation of proposed modifications to MPE for large scale discriminative training," in *Proc. ICASSP*, 2007, vol. 4, pp. 321–324.

[3] Zheng Chen, Kai-Fu Lee, and Ming-Jing Li, "Discriminative training on language model," in *Proc. ICSLP*, 2000.

[4] Hong-Kwang Jeff Kuo, Eric Fosler-Lussier, Hui Jiang, and Chin-Hui Lee, "Discriminative training of language models for speech recognition," in *Proc. ICASSP*, 2002, vol. 1, pp. 325–328.

[5] Jen-Wei Kuo and Berlin Chen, "Minimum word error based discriminative training of language models," in *Proc. INTERSPEECH*, 2005, pp. 1277–1280.

[6] Brian Roark, Murat Saraclar, and Michael Collins, "Discriminative n-gram language modeling," *Computer Speech & Language*, vol. 21(2), pp. 373–392, 2006.

[7] Shiuan-Sung Lin and François Yvon, "Discriminative training of finite-state decoding graphs," in *Proc. INTERSPEECH*, 2005, pp. 733–736.

[8] Hong-Kwang Jeff Kuo, Brian Kingsbury, and Geoffrey Zweig, "Discriminative training of decoding graphs for large vocabulary continuous speech recognition," in *Proc. ICASSP*, 2007, vol. 4, pp. 45–48.

[9] Michiel Bacchiani, Françoise Beaufays, Johan Schalkwyk, Mike Schuster, and Brian Strope, "Deploying GOOG-411: early lessons in data, measurement, and testing," in *Proc. ICASSP*, 2008, pp. 5260–5263.

[10] Geoffrey Zweig, Olivier Siohan, George Saon, Bhuvana Ramabhadran, Daniel Povey, Lidia Mangu, and Brian Kingsbury, "Automated quality monitoring in the call center with asr and maximum entropy," in *Proc. ICASSP*, 2006, vol. 1, pp. 589–592.

[11] Harry Printz and Peder A. Olsen, "Theory and practice of acoustic confusability," *Computer Speech & Language*, vol. 16(1), pp. 131–164, 2002.

[12] Binit Mohanty, John R. Hershey, Peder A. Olsen, Suleyman Kozat, and Vaibhava Goel, "Optimizing speech recognition grammars using a measure of similarity between hidden Markov models," in *Proc. ICASSP*, 2008, pp. 4953–4956.

[13] Nobuaki Minematsu, Gakuto Kurata, and Keikichi Hirose, "Integration of MLLR adaptation with pronunciation proficiency adaptation for non-native speech recognition," in *Proc. ICSLP*, 2002, pp. 529–532.

[14] Mehryar Mohri, Fernando Pereira, and Michael Riley, "Weighted finite-state transducers in speech recognition," *Computer Speech & Language*, vol. 16(1), pp. 69–88, 2002.

[15] Stanley F. Chen and Joshua Goodman, "An empirical study of smoothing techniques for language modeling," *Computer Speech and Language*, vol. 13, no. 4, pp. 359–393, 1999.

[16] Jonathan Le Roux and Erik McDermott, "Optimization method for discriminative training," in *Proc. INTERSPEECH*, 2005, pp. 3341–3344.

[17] Jorge Silva and Shrikanth Narayanan, "Average divergence distance as a statistical discrimination measure for hidden Markov models," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14(3), pp. 890–906, 2006.

[18] John R. Hershey and Peder A. Olsen, "Approximating the Kullback Leibler divergence between gaussian mixture models," in *Proc. ICASSP*, 2007, vol. 4, pp. 317–320.

[19] John R. Hershey and Peder A. Olsen, "Variational bhattacharyya divergence for hidden Markov models," in *Proc. ICASSP*, 2008, pp. 4557–4560.