WFST-BASED STRUCTURAL CLASSIFICATION INTEGRATING DNN ACOUSTIC FEATURES AND RNN LANGUAGE FEATURES FOR SPEECH RECOGNITION

Quoc Truong Do**, Satoshi Nakamura***, Marc Delcroix*†*, Takaaki Hori*†

* Nara Institute of Science and Technology, Nara Japan
{do.truong.dj3, s-nakamura}@is.naist.jp
† NTT Communication Science Laboratories, Kyoto, Japan
{hori.t, marc.delcroix}@lab.ntt.co.jp

ABSTRACT

This paper proposes a method to train Weighted Finite State Transducer (WFST) based structural classifiers using deep neural network (DNN) acoustic features and recurrent neural network (RNN) language features for speech recognition. Structural classification is an effective approach to achieve highly accurate recognition of structured data in which the classifier is optimized to maximize the discriminative performance using different kinds of features. A WFST-based classifier, which can integrate acoustic, pronunciation, and language features embedded in a composed WFST, was recently extended to incorporate DNN bottleneck (DNNBN) features. In this paper, we further investigate the integration of a RNN language model (RNNLM) with the WFST classifier. To this end, we introduce a lattice rescoring method using a RNNLM for efficient classifier training. In a lecture transcription task, we reduced the word error rate from 19.2% to 18.6% by optimizing the WFST parameters for the DNNBN acoustic and RNNLM language features.

***Index Terms*—** Structural classification, WFST-DNN, RNNLM, Lattice rescoring, Speech recognition

1. INTRODUCTION

Structural classification [1] is an effective approach to achieve highly accurate recognition of structured data in which the classifier is optimized to maximize the discriminative performance using different kinds of features. For instance, conditional random fields (CRFs) with latent variables, called hidden CRFs, were applied to phone recognition as a structural classification problem [2, 3]. A segmental CRF framework was proposed that can integrate several multi-scale detector streams as input features for speech recognition [4].

The WFST-based classifier [5, 6, 7] for speech recognition, which is one of the most successful examples of structural classification, can integrate acoustic, pronunciation, and language features embedded in a composed WFST to provide accurate speech recognition. It was recently extended to incorporate DNN bottleneck (DNNBN) acoustic features [8, 9]. The extended classifier, called WFST-DNN, yields a successful way to integrate WFST-based speech recognition and DNN feature extraction into a unified framework. It inherits the advantages of a DNN-based feature extractor [10] and the rich representation of WFST that describes in a unique graph all the linguistic information, such as a language model, a lexicon and the phoneme context dependence. Its advantage is that it can model the interdependence of acoustic and linguistic aspects and optimize the parameters based on a single objective function. Its performance has been evaluated, and it outperformed standard speech recognition based on GMM-HMM or DNN-HMM [8].

Recently, several investigations of continuous space language models [11, 12, 13, 14] have shown that they handle the data sparsity problem better than n-gram language models. Among various types of continuous space language models, the recurrent neural network language model (RNNLM) is an effective method that significantly improves language modeling and speech recognition. Previous studies [13, 14] showed that the accuracy of speech recognition was significantly improved, especially when the RNNLM was interpolated with a standard n-gram model. An advantage of RNNLM is that it can capture long context patterns of entire utterances rather than just $n-1$ previous words in the n-gram models. On the other hand, RNNLM increases the computational complexity in decoding because of its long history dependency. However, efficient lattice rescoring [15] and one-pass decoding [16, 17] have been proposed recently.

In this paper, we integrate RNNLM into structural classification approaches instead of the standard n-gram language models that are conventionally used. Looking at the effectiveness of RNNLM, we expect that further WER reduction can be obtained by introducing it into the structural classification approach. For example, it is possible to simply combine RNNLM with an already trained WFST-DNN in the decoding phase. However, the parameters of WFST-DNN have been optimized for the n-gram language model but not for RNNLM. In this paper, we present a method that optimizes a WFST-DNN structured classifier with RNNLM. Our approach integrates the RNNLM probabilities into the training lattices and performs discriminative training to optimize the WFST-DNN parameters. To this end, we introduce an effective lattice rescoring method that serves as an important step of the optimization process.

2. WFST-DNN CLASSIFIER FOR SPEECH RECOGNITION

A WFST-DNN classifier [9, 8] is a unified framework that combines a WFST-based classifier and a DNN-based feature extractor.

In decoding with a WFST-DNN classifier, the target is to find the most plausible word sequence $\hat{\ell}$ given an input sequence of speech features $\mathbf{X} = \{x_1, x_2, \dots\}$ by searching for the most plausible arc sequence $\hat{\mathbf{a}} = \{\hat{a}_1, \hat{a}_2, \dots\}$. The output word sequence is extracted by concatenating the output symbols of the arc sequence in the WFST as:

$$\hat{\ell} = O[\hat{\mathbf{a}}], \tag{1}$$

$$\hat{\mathbf{a}} = \underset{\mathbf{a}\in\mathcal{D}}{\operatorname{argmax}}\, P(\mathbf{a}|\mathbf{X}), \tag{2}$$

where $O[\hat{\mathbf{a}}]$ denotes the output symbol sequence corresponding to arc sequence $\hat{\mathbf{a}}$. $\mathcal{D}$ is a decoding graph, which is the composition of C WFSTs, i.e., $\mathcal{D} = \mathcal{D}_1 \circ \mathcal{D}_2 \circ \ldots \mathcal{D}_C$. Hence, each arc a_n in $\mathcal{D}$ can be represented as a tuple of arcs as $a_n \equiv (a_n^{(1)}, a_n^{(2)}, \ldots, a_n^{(C)})$. The posterior probability of arc sequence $P(\mathbf{a}|\mathbf{X})$ is defined as

$$P(\mathbf{a}|\mathbf{X}) \propto exp\left\{\sum_n -\omega(a_n;\mathbf{X})\right\}. \quad (3)$$

In this equation, arc cost $\omega(a_n;\mathbf{X})$ can be expressed as follows:

$$\omega(a_n;\mathbf{X}) = g(a_n^{(1)};\mathbf{X}) + \sum_{c=2}^{C} r_c W[a_n^{(c)}], \quad (4)$$

where $W[a_n^{(c)}]$ is the transition weight corresponding to arc $a_n^{(c)}$, and r_c is a scaling factor for the c-th WFST. Transition weight function $g(a_n^{(1)};\mathbf{X})$ is defined by

$$g(a_n^{(1)};\mathbf{X}) = \gamma_{a_n^{(1)}} + \sum_{\tau=T[a_n^{(1)}]}^{T'[a_n^{(1)}]} ((\boldsymbol{\alpha}_{a_n^{(1)}})^\intercal \mathbf{h}^{(L)}(x_\tau) + \beta_{a_n^{(1)}}), \quad (5)$$

where x_τ is the τ-th feature vector in input feature sequence $\mathbf{X}$, $T[a]$ and $T'[a]$ indicate the beginning and ending time frames for arc a, respectively. $\mathbf{h}^{(L)}(x_\tau)$ is the activation vector of the last hidden layer of the DNN. $\boldsymbol{\alpha}_{a_n^{(1)}}$, $\beta_{a_n^{(1)}}$, and $\gamma_{a_n^{(1)}}$ are the parameters of WFST-DNN to be optimized, where $\boldsymbol{\alpha}_{a_n^{(1)}}$ is the weight vector corresponding to arc $a_n^{(1)}$ in the weight matrix between the last hidden layer and the output layer of the DNN. $\beta_{a_n^{(1)}}$ is the bias parameter of the output layer. $\gamma_{a_n^{(1)}}$ is the transition weight of arc $a_n^{(1)}$, which is equivalent to $r_1 W[a_n^{(1)}]$ but is also jointly optimized. The unified WFST-DNN framework is illustrated in Fig. 1.

The advantage of WFST-DNN is that it can model both acoustic and language aspects in a unified form. Hence, joint sequence training of the acoustic and language models can be achieved by discriminative training [9]. Note that, in large vocabulary tasks, it is difficult to use a single WFST composed up to the n-gram language model, because the size of the WFST increases and an enormous number of parameters have to be estimated. Practically, we employ a composition of models up to the unigram language model as the first WFST, and the n-gram language model as the second WFST in Eq. (4). In this case, we can assume that parameter tying is enforced for the features beyond the word unigram. Even with this parameter tying, the WFST-DNN yielded significant improvements in recognition accuracy [9].

In the latest setting of WFST-DNN training, the model parameters have been optimized with the n-gram language model. Since RNNLM has been shown to be an effective way for language modeling, it is promising to optimize the model parameters with RNNLM.

3. RECURRENT NEURAL NETWORK LANGUAGE MODELS

In this work, we adopt a class-based RNNLM [13], whose topology is shown in Fig. 2 with three layers: input, hidden, and output layers. The input vector for the i-th word is a combination of the word vector in the 1-of-N coding and the previous hidden layer activation:

$$\mathbf{x}_i = [\mathbf{w}_{i-1}^\intercal, \mathbf{s}_{i-1}^\intercal]^\intercal. \quad (6)$$

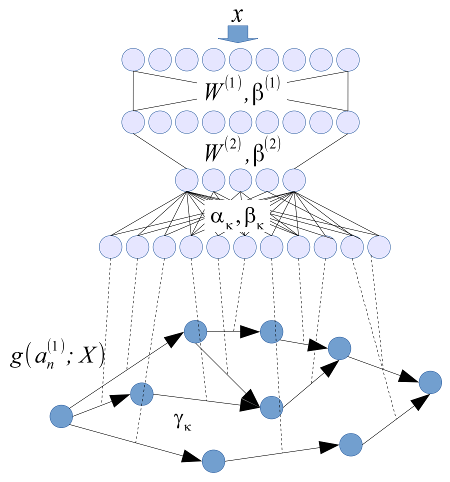

Fig. 1. Unified WFST-DNN classifier

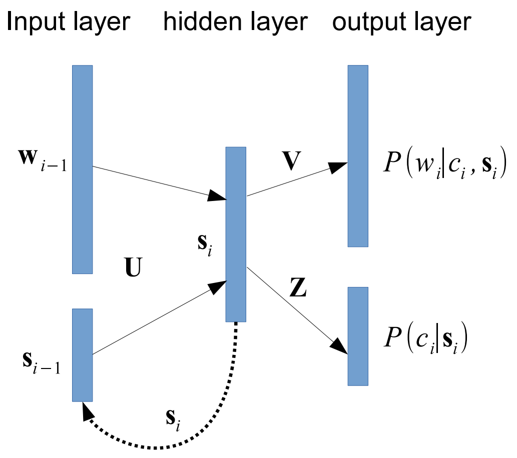

Fig. 2. A class-based RNNLM

The advantage of class-based RNNLMs is that they reduce the computation cost. Each word in the output layer is assigned to a unique class based on the frequency counts in the training data. The probability of w_i, which is the i^{th} word in class c_i, is calculated as

$$P(w_i|\mathbf{x}_i) = P(w_i|c_i, \mathbf{s}_i)P(c_i|\mathbf{s}_i). \quad (7)$$

The RNNLM is converted to the WFST form on-the-fly by a previously proposed method in [17]. In this paper, the transition weights in the RNNLM-WFST are computed as the interpolation of the RNNLM and n-gram model as

$$P_{comb}(w|p) = \lambda P_{rnn}(w|p) + (1-\lambda)P_{ngram}(w|\phi(p)), \quad (8)$$

where λ is the interpolation factor and $\phi(p)$ represents the n-1 word history for state p in the WFST.

4. OPTIMIZATION OF WFST-DNN WITH RNNLM PROBABILITIES

In this section, we describe the process that optimizes the set of model parameters, $\Lambda = \{\boldsymbol{\alpha}_\kappa, \beta_\kappa, \gamma_\kappa | \kappa = 1, \ldots, A\}$ with RNNLM probabilities, where A is the number of arcs in the WFST. The training process is an extension of a previous work [8] that uses minimum transition error training methods [18].

The objective function to be maximized is defined based on the boosted MMI criterion [19]:

$$F_{\sigma}^{bMMI}(\Lambda) = \sum_{j} \log \frac{\exp\{-\Omega(\mathbf{X}_j, \mathbf{a}_j)\}}{\sum_{\mathbf{a}' \in \mathcal{L}_j} \exp\{-\Omega(\mathbf{X}_j, \mathbf{a}') + \sigma E(\mathbf{a}_j, \mathbf{a}')\}}, \tag{9}$$

where $\mathbf{X}_j$ is the j-th utterance in the training corpus, and $\mathbf{a}_j$ and $\mathcal{L}_j$ are the reference and recognition lattices for $\mathbf{X}_j$, respectively. $E(\mathbf{a}, \mathbf{a}')$ is the transition error count function, and σ is a boosting parameter. $\Omega(\mathbf{X}, \mathbf{a}')$ is the total cost defined as

$$\Omega(\mathbf{X}, \mathbf{a}) = \sum_{n} -\omega(a_n; \mathbf{X}). \tag{10}$$

In the standard WFST-DNN training for large vocabulary tasks, transition weight $W[a_n^{(c)}]$, which is used to compute the cost function in Eq. (4), is given by an n-gram language model. To optimize the WFST-DNN parameters with RNNLM, we need to apply the RNNLM probabilities of Eq. (8) to the transition weight by rescoring the training lattices, $\mathbf{a}_j$ and $\mathcal{L}_j$. However, the RNNLM requires the entire history from the beginning of the utterances to calculate the probability for a given word. Hence, the rescored lattices grow exponentially with the length of the utterances. Therefore, efficient lattice rescoring is needed to generate appropriate RNNLM rescored lattices for training[1]. In the next section, we describe lattice rescoring with RNNLM that generates rescored lattices with small size while keeping better paths in the lattices.

5. EFFICIENT LATTICE RESCORING

As described in the previous section, RNNLM rescored lattices are required to train WFST-DNN. In this section, we describe the lattice rescoring method. The application of this method is limited not only to generate lattices for WFST-DNN training but also for standard 2-pass decoding in speech recognition.

The strong generalization of RNNLM is derived from a full context history that is represented by a continuous history vector. For this reason, the number of RNNLM contexts grows exponentially as the hypotheses become longer. A solution is to develop a history clustering context to allow multiple states to share the same history. A history vector-based clustering method was proposed [15] that generates reasonably well rescored lattices. However, the previous work used approximated RNNLMs, where histories are clustered in advance. Since such approximation decreases the recognition accuracy, it cannot be used to train WFST-DNN where we use exact (non-approximated) RNNLM.

To address the above problem, we propose a history vector clustering method that takes the accumulated weight into account. The idea is to give priority to all the states based on their accumulated weight from the initial state, where a state with a smaller accumulated weight is assumed to be better, i.e., it has higher priority. The states are then expanded based on their priority. Figure 3 illustrates the expansion process. The expanded state candidates 2', 2", and 2"' are derived from the same state 2 in the original lattice. If the state 2' has the smallest accumulated weight, it is expanded first. When the state clustering is performed, the state with the larger accumulated

[1]In our approach, the RNNLM itself is not optimized but the transition weight functions are optimized based on the entire scores including the RNNLM probabilities.

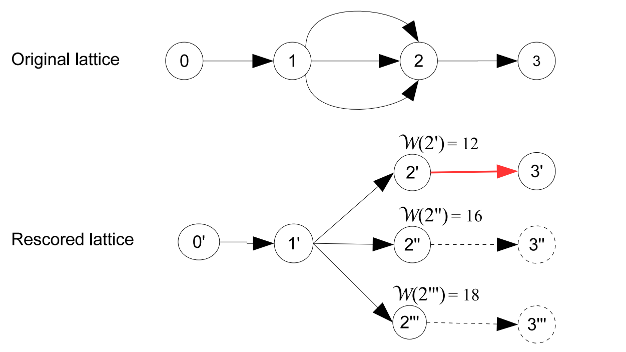

Fig. 3. Expansion based on accumulate priority. State 2' is expanded first because it has the smallest accumulated weight among other state candidates.

weight will be merged into the smaller one. This process ensures that good paths remain in the rescored lattices.

We denote $\mathcal{W}(s)$ is the accumulated weight from the initial state to state s in a lattice being generated by RNNLM rescoring, which is calculated as

$$\mathcal{W}(s) = \mathcal{W}(s') + W[a_{s',s}], \tag{11}$$

where s' is a preceding state of s and $a_{s',s}$ is an arc leaving s' to s. $W[a_{s's}]$ is the transition weight of the arc.

The history vector of state s is given as a hidden activation vector in the RNNLM, which is computed with the word sequence on the best path to state s. The history distance of given two states, s_1 and s_2, is calculated by the Euclidean distance:

$$D(s_1, s_2) = \sqrt{\sum_{k=1}^{K} (h_{s_1}^{(k)} - h_{s_2}^{(k)})^2}, \tag{12}$$

where $h_s^{(k)}$ denotes the k-th element of the K-dimensional history vector of state s. States that have a history distance smaller than a threshold from each other will be merged. Note that when similar states are merged, the arcs connected to one of the similar states are reconnected to the merged state. To control the size of the rescored lattices, we also introduce *state beam width* m, which is the maximum number of states derived from each original state. When the number of states reaches the m value, the remaining states with lower priority are merged to the closest state based on the history distance. With this rescoring strategy, we can reduce the size of lattices while keeping better paths.

6. RELATION TO PRIOR WORK

There have been attempts to introduce additional features in structural classification approaches for speech recognition. Segment-based features such as the duration information of phonemes were introduced [20]. Example-based features based on the DTW distance from the k-nearest neighbor speech examples were also investigated [21]. However, no prior work has considered RNNLM probabilities as presented in this paper.

In addition, we created a lattice rescoring algorithm, which can assign correct RNNLM probabilities to multiple hypotheses with relatively high scores in a lattice, to compute the objective function more correctly. A lattice rescoring method was proposed but it makes several approximations that may affect the performance [15].

Since our speech recognizer can perform one-pass decoding using a RNNLM without approximation [17], we needed another lattice rescoring method for a non-approximated RNNLM.

On the other hand, sequence training methods have recently been applied to DNN acoustic models [22, 23, 24]. Those methods employ MMI or minimum Bayes risk objective function. The WFST-DNN is also optimized by sequence training, but the transition weight functions of acoustic, pronunciation and language model features are optimized. Although only the parameters in the output layer of the DNN get updated in this training step, sufficient error reduction can be obtained as reported in [24].

7. EXPERIMENTS

In our experiments, we evaluated the RNNLM lattice rescoring method and the WFST-DNN optimized with the rescored lattices. We conducted experiments using the MIT lecture corpus [25] with 101 hours in the training set, 54 minutes in the development set and 7.8 hours in the evaluation set. As input feature vectors of a DNN, we extracted 12 Mel-frequency cepstral coefficients and a log-energy feature augmented by their first- and second-order derivatives, and then the 38 dimensional features excluding the static log-energy were further augmented by concatenating the feature vectors of the 5 preceding and 5 subsequent frames. For the baseline DNN-HMM system, a DNN was set up with 7 layers, in which 418 units for the input layer, 2048 units for each hidden layer, and 512 units for the bottleneck layer were prepared. The output layer consisted of 2,565 units, which equals the number of HMM states. The DNN parameters were trained using a cross-entropy criterion. In the WFST-DNN system, we used two WFSTs, where $\mathcal{D}_1$ was composed of the HMM state, the triphone context dependency and the lexicon WFSTs, and $\mathcal{D}_2$ was generated from either a trigram language model with Kneser-Ney smoothing or a RNNLM linearly combined with the trigram model. The two WFSTs were combined on-the-fly during decoding [16]. The vocabulary size of the lexicon and the language models was 44,485. We employed a class-based RNNLM with 300 hidden units and 250 word classes trained with a sentence-independent condition. The input and hidden layers of the WFST-DNN were the same as the DNN-HMM system, but the output layer consisted of 108,485 units, which equals the number of arcs in the WFST $\mathcal{D}_1$. All the WFST-DNN parameters were trained with bMMI or MPE criterion.

7.1. Evaluation of RNNLM lattice rescoring

We first evaluated the RNNLM lattice rescoring method. Word error rate (WER) for the best path in each lattice and its lattice density were obtained for the development set. The lattice density was computed as the ratio of the number of arcs in the recognition lattices to that in the reference lattices. The results are shown in Table 1.

The original lattices were generated using the DNN-HMM acoustic model and the trigram language model, for which we obtained 28.2% WER and 259.7 lattice density. The original lattices were then rescored with the RNNLM. We also tested one-pass decoding with the RNNLM using the method in [17], and obtained 26.9% WER. In [17], this method achieved a better accuracy than that by 1000-best rescoring. Accordingly, the WER is considered as almost the lower limit obtained by the RNNLM.

In lattice rescoring, we changed two parameters: distance threshold t and state beam width m. Looking at the results, they are promising since our rescoring method can produce exactly the same WER with one-pass decoding, even though the lattice density is slightly higher than the original lattices in many cases. Thus, our proposed lattice rescoring method hopefully assign correct accumulated weights to better hypotheses in each lattice, and therefore can be used for WFST-DNN training.

Table 1. RNNLM rescoring performance

System	WER	Lattice density
Original lattices	28.2	259.7
one-pass [17]	**26.9**	-
t=2, m=∞	**26.9**	276.3
t=1.5, m=∞	**26.9**	303.7
t=1.3, m=∞	**26.9**	332.2
t=1, m=∞	**26.9**	456.9
t=0, m=1	**26.9**	259.7
t=0, m=5	**26.9**	1189.9
t=1, m=5	**26.9**	379.7
t=2, m=5	**26.9**	275.7

Table 2. Word error rate for MIT lecture transcription

System	Dev.	Eval.
DNN-HMM w/ 3-gram	28.3	22.4
DNN-HMM w/ RNNLM	26.9	20.2
WFST-DNN w/ 3-gram	26.3	20.6
WFST-DNN w/ RNNLM	24.9	19.2
Optimized (proposed)	**24.6**	**18.6**
WFST-DNN w/ 3-gram (MPE)	26.3	20.7
WFST-DNN w/ RNNLM (MPE)	25.0	19.3
Optimized (proposed) (MPE)	**24.5**	**18.8**

7.2. Evaluation of WFST-DNN optimized with RNNLM

WFST-DNN is optimized with RNNLM by discriminative training with the RNNLM rescored lattices. We chose $t = 1$ and $m = 5$ for rescoring training lattices. Table 2 shows the word error rate in speech recognition using the standard DNN-HMM system or WFST-DNN classifier, in which the trigram LM and/or the RNNLM was used in one-pass decoding. Compared to the DNN-HMM baseline, the RNNLM and WFST-DNN reduced the WER from 22.4% to 20.2% and 20.6%, respectively. The simple combination of WFST-DNN and RNNLM also reduced the WER to 19.2%. Finally, the proposed approach that optimized the WFST-DNN using RNNLM rescored lattices achieved 18.6%, which is significant error reduction from 19.2%. In addition, we tested the optimization step with the MPE criterion based on differenced MMI (dMMI) [26]. We also observed substantial improvements, as in the case of bMMI.

8. CONCLUSION

In this paper, we described a method that integrates WFST-based structured classifiers with a DNN acoustic model and a RNN language model. We also described an effective RNNLM rescoring method that produces appropriate rescored lattices for WFST-DNN training. The WFST-DNN that was optimized with RNNLM probabilities yields a better performance than the standard WFST-DNN optimized only for trigram probabilities [8]. Future work will include joint training of WFST-DNN and RNNLM parameters.

9. REFERENCES

[1] M. Gales, S. Watanabe, and E. Fosler-Lussier, "Structured discriminative models for speech recognition: An overview," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 70–81, 2012.

[2] A. Gunawardana, M. Mahajan, A. Acero, and J. C. Platt, "Hidden conditional random fields for phone classification," in *Proceedings of Interspeech*, 2005, pp. 1117–1120.

[3] Y. H. Sung and D. Jurafsky, "Hidden conditional random fields for phone recognition," in *Proceedings of ASRU*, 2009, pp. 107–112.

[4] G. Zweig and P. Nguyen, "A segmental crf approach to large vocabulary continuous speech recognition," in *Proceedings of ASRU*, 2009, pp. 152–157.

[5] S. Watanabe, T. Hori, and A. Nakamura, "Large vocabulary continuous speech recognition using WFST-based linear classifier for structured data," in *Proceedings of Interspeech*, 2010, pp. 346–349.

[6] A. Lehr and I. Shafran, "Learning a discriminative weighted finite-state transducer for speech recognition," *IEEE Trans. Audio Speech Lang. Processing*, vol. 21, pp. 1360–1367, 2011.

[7] Y. Kubo, S. Watanabe, T. Hori, and A. Nakamura, "Structural classification methods based on weighted finite-state transducers for automatic speech recognition," *IEEE Trans. Audio Speech Lang. Processing*, vol. 20, pp. 2240–2250, 2012.

[8] Y. Kubo, T. Hori, and A Nakamura, "Integrating deep neural networks into structured classification approach based on weighted finite-state transducer," in *Proceedings of INTERSPEECH*, 2012.

[9] Y. Kubo, T. Hori, and A Nakamura, "Large vocabulary continuous speech recognition based on WFST structured classifiers and deep bottleneck features," in *Proceedings of ICASSP*, May 2013, pp. 7629–7633.

[10] D. Yu and M. Seltzer, "Improved bottleneck features using pretrained deep neural networks," in *Proceedings of Interspeech*, August 2011.

[11] H. Schwenk, "Continuous space language models," *Computer Speech and Language*, vol. 21, pp. 492–518, 2007.

[12] T. Mikolov, M. Karafiat, L. Burget, J. Cernocky, and S. Khudanpur, "Recurrent neural network based language model," in *Proceedings of Interspeech*, 2010, pp. 1045–1048.

[13] T. Mikolov, S. Kombrink, L. Burget, J. Cernocky, and S. Khudanpur, "Extensions of recurrent neural network language model," in *Proceedings of ICASSP*, 2011, pp. 5528–5531.

[14] S. Kombrink, M. Karafiat T. Mikolov, and L. Burget, "Recurrent neural network based language modeling in meeting recognition," in *Proceedings of Interspeech*, 2011, pp. 2877–2880.

[15] X. Liu, Y. Wang, X. Chen, M.J.F. Gales, and P.C. Woodland, "Efficient lattice rescoring using recurrent neural network language models," in *Proceedings of ICASSP*, 2014.

[16] T. Hori, C. Hori, Y. Minami, and A. Nakamura, "Efficient WFST-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, pp. 1352–1365, 2007.

[17] T. Hori, Y. Kubo, and A. Nakamura, "Real-time one-pass decoding with recurrent neural network language model for speech recognition," in *Proceedings of ICASSP*, 2014.

[18] Y. Kubo, S. Watanabe, and A Nakamura, "Decoding network optimization using minimum transition error training," in *Proceedings of ICASSP*, 2012, pp. 4197–4200.

[19] D. Povey, D. Kanevsky, B. Kingsbury, B. Ramabhadran, G. Saon, and K. Visweswariah, "Boosted MMI for model and feature-space discriminative training," in *Proceedings of ICASSP*, 2008, pp. 4057–4060.

[20] G. Zweig, P. Nguyen, D. V. Compernolle, K. Demuynck, L. Atlas, P. Clark, G. Sell, M. Wang, F. Sha, H. Hermansky, D. Karakos, A. Jansen, S. Thomas, G.S.V.S. Sivaram, S. Bowman, and J Kao, "Speech recognition with segmental conditional random fields: A summary of the JHU CLSP 2010 summer workshop," in *Proceedings of ICASSP*, 2011.

[21] K. Demuynck, D. Seppi, D.V. Compernolle, P. Nguyen, and G. Zweig, "Integrating meta-information into exemplar-based speech recognition with segmental conditional random fields," in *Proceedings of ICASSP*, 2011.

[22] B. Kingsbury, T. N. Sainath, and H. Soltau, "Scalable minimum Bayes risk training of deep neural network acoustic models using distributed hessian-free optimization," in *Proceedings of Interspeech*, 2012.

[23] K. Vesely, A. Ghoshal, L. Burget, and D. Povey, "Sequence-discriminative training of deep neural networks," in *Proceedings of Interspeech*, 2013, pp. 2345–2349.

[24] G. Heigold, E. McDermott, V. Vanhoucke, A. Senior, and M. Bacchiani, "Asynchronous stochastic optimization for sequence training of deep neural networks," in *Proceedings of ICASSP*, 2014, pp. 5624–5628.

[25] J. Glass, T.J. Hazen, S. Cyphers, I. Malioutov, D. Huynh, and R.Barzilay, "Recent progress in the mit spoken lecture processing project," in *Proceedings of Interspeech*, 2007, pp. 2553–2556.

[26] E. McDermott, S. Watanabe, and A Nakamura, "Discriminative training based on an integrated view of MPE and MMI in margin and error space," in *Proceedings of ICASSP*, 2010, pp. 4894–4897.