GRAPH-BASED SEMI-SUPERVISED ACOUSTIC MODELING IN DNN-BASED SPEECH RECOGNITION

Yuzong Liu, Katrin Kirchhoff

Department of Electrical Engineering
University of Washington
Seattle, WA 98195, USA

ABSTRACT

This paper describes the combination of two recent machine learning techniques for acoustic modeling in speech recognition: deep neural networks (DNNs) and graph-based semi-supervised learning (SSL). While DNNs have been shown to be powerful supervised classifiers and have achieved considerable success in speech recognition, graph-based SSL can exploit valuable complementary information derived from the manifold structure of the unlabeled test data. Previous work on graph-based SSL in acoustic modeling has been limited to frame-level classification tasks and has not been compared to, or integrated with, state-of-the-art DNN/HMM recognizers. This paper represents the first integration of graph-based SSL with DNN based speech recognition and analyzes its effect on word recognition performance. The approach is evaluated on two small vocabulary speech recognition tasks and shows a significant improvement in HMM state classification accuracy as well as a consistent reduction in word error rate over a state-of-the-art DNN/HMM baseline.

Index Terms— Acoustic modeling, deep neural networks, graph-based learning, semi-supervised learning

1. INTRODUCTION

Among recent attempts at improving acoustic modeling in automatic speech recognition (ASR), approaches based on deep neural networks (DNNs) have achieved considerable success. DNNs are supervised classifiers consisting of a multi-layered neural networks whose layers are typically first trained independently and are subsequently optimized jointly. In ASR they are used to map from acoustic feature vectors to probability distributions over hidden Markov model (HMM) states.

Despite the success of DNN/HMM based system it is worth exploring additional, complementary machine learning techniques that might further improve acoustic modeling. In this paper we investigate the combination of DNN/HMM based speech recognition and semi-supervised learning (SSL), in particular its variant graph-based learning (GBL). In GBL the labeled training data and the unlabeled test data are jointly represented as a graph, whose nodes represent data samples and whose edge weights encode pairwise similarities between samples. GBL assigns labels to the unlabeled data points under the smoothness constraints imposed by the graph: two samples related by a highly weighted edge are likely to receive the same label, while dissimilar samples should receive dissimilar classification outputs. By exploiting smoothness constraints we can often achieve more accurate classifications as well as an implicit adaptation to the test data. Note that in addition to training-test sample similarities, GBL also utilizes similarities between different test samples, which is a fundamental difference to standard supervised learning techniques that classify each test sample independently. GBL has been successfully used for a variety of machine learning problems and has been applied to simple speech processing tasks in [1, 2, 3, 4, 5]. However, all of these previous works suffer from two shortcomings: (a) They only focused on the relatively simple task of frame-based phone classification or phone segment classification but did not explore fully-fledged word recognition. (b) The baseline classifiers used in these papers were GMMs or GMM/HMM systems. These have since been superseded by DNN/HMM based systems; however, it is not clear whether GBL can also outperform state-of-the-art DNN classifiers.

This study makes the following contributions: First, we combine graph-based SSL with state-of-the-art DNN/HMM systems and compare several ways of constructing data graphs that are directly dependent on the architecture of the DNNs we use. Second, we investigate for the first time the effect of GBL on word error rate (WER). We demonstrate that the GBL-enriched DNN/HMM systems achieve consistent reductions in WER over the baseline systems. Finally, we compare GBL against self-training, which also utilizes unlabeled test data. We find that a single application of GBL achieves improvements comparable to those obtained by several iterations of self-training.

2. GRAPH-BASED SEMI-SUPERVISED LEARNING

Let $D_L = \{x_i, r_i\}_{i=1}^{l}$ be the set of labeled samples, and $D_U = \{x_i\}_{l+1}^{l+u}$ be the set of unlabeled samples; r_i encodes

the label distribution of labeled samples. Let l and u be the number of labeled samples and unlabeled samples respectively, and $n = l + u$ be the total number of samples. Traditional supervised classification *only* utilizes the information from D_L to learn a decision boundary, whereas in a semi-supervised transductive learning framework, the decision boundary is obtained by utilizing the information from *both* D_L and D_U at the same time. GBL models the joint data $D = D_L \cup D_U$ as a graph $G = (V, E)$ consisting of a set of nodes V representing the samples, and a set of weighted edges E representing similarities between samples. Typically, a k nearest neighbor (kNN) graph is used. Several graph-based learning algorithms have been proposed [6, 2, 7, 4, 5] that utilize smoothness over the graph neighborhood as a regularization term in their objective function. In this work we adopt **prior-regularized measure propagation (pMP)** [2, 5] as our graph-based learning algorithm. We choose pMP over other GBL algorithms, because they emperically show consistent better performance on phonetic classification tasks [8, 5]. The pMP algorithm minimizes the following objective function:

$$\begin{aligned} F(D, G, \tilde{p}) \quad = \quad & \sum_{i=1}^{l} KL(r_i||p_i) + \mu \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} w_{ij} KL(p_i||p_j) \\ & + \nu \sum_{i=l+1}^{l+u} KL(p_i||\tilde{p}_i) \end{aligned} \quad (1)$$

where r and p are the true and hypothesized probability distributions over labels, respectively, and $\tilde{p}$ is a prior label distribution. $KL(p, q)$ is the Kullback-Leibler divergence between distributions p and q, $\mathcal{N}_i$ is the graph neighborhood of sample i (i.e., the set of nodes directly connected to it), and w_{ij} is the weight of the edge linking samples i and j. μ and ν are weighting coefficients that are optimized on the development set. The first term in Equation 1 measures the divergence between the true and predicted label distributions on the training samples. The second term measures the smoothness of the predicted distributions within graph neighborhoods: samples that have a larger similarity weight w_{ij} are encouraged to have similar label distributions. The last term ensures that the inferred distribution does not deviate too strongly from a prior distribution, which can be either uniform or provided by another, external source of information (such as a supervised classifier trained a priori). This objective function can be solved efficiently via alternating minimization. The outcome is the set of predicted probability distributions on the unlabeled data points. The graph itself can be constructed in a number of different ways. Here, we use an RBF kernel function

$$w_{ij} = \exp(-\frac{d_{ij}}{\sigma}) \quad (2)$$

where d_{ij} is the ℓ_2 distance between i and j, and σ is a bandwidth parameter ($\sigma = 500$). The graph is then built according to Algorithm 1.

Algorithm 1 Graph construction

1: Given: labeled set D_L, unlabeled set D_U, and similarity measure w
2: **for** $i = l + 1$ to $l + U$ **do**
3: Find k nearest neighbors for sample i in D_L, with their scaled similarity weights $a \cdot w$;
4: Find k nearest neighbors for sample i in D_U, with their scaled similarity weights $b \cdot w$;
5: **end for**
6: Enforce the weights on the graph to be symmetric: $w'_{ij} = \max(w_{ij}, w_{ji})$
7: **return** Final graph weight matrix $\mathbf{W} = \{w'_{ij}\}, \forall i, j, 1 \leq i, j \leq l + u$.

3. SYSTEM INTEGRATION

In this section, we explain the framework of integrating GBL into a fully-fledge DNN-based system.

Step 1 - Supervised DNN training: We first train a DNN using the training samples. This DNN is then run on the test samples to create HMM state prediction, which is used as $\{\tilde{p}_i\}$ in Equation 1. The training sample distributions for the first term in Equation 1 are created from the state-level forced alignment.

Step 2 - Graph construction: We create a graph over the entire training and test samples. Different choices of feature representation for each node are available and they are further discussed in Section 5.1. The number of neighbors per node is limited to $k = 10$. We use the ANN library [1] for fast, approximate kNN search. The scaling factors a, b for similarity weights (see Algorithm 1) are tuned on the development set; we use $a = 1, b = 5$.

Step 3 - GBL: We apply GBL in this step to produce a changed posterior probability distribution for each HMM state s given acoustic feature vector $\mathbf{x}$, $p(s|\mathbf{x})$ on the test samples. Just as in the baseline DNN/HMM system, we convert the posteriors into scaled likelihoods $p(\mathbf{x}|s)$ by dividing by the prior distribution over HMM states, $p(s)$.

Step 4 - Lattice rescoring: These new likelihoods are then linearly interpolated with the original acoustic score and the LM score in the original lattices. The interpolation weights are optimized on the development set.

4. DATA AND SYSTEMS

We evaluate our approach on two datasets. The first is the DARPA Resource Management (RM) task. For training we use the RM-SI dataset, which consists of 3990 sentences. We use the *test-feb89* set as our dev set, which consists of 300 sentences. As eval sets we use *test-oct89*, *test-feb91*, and *test-sep92*, each of which consists of 300 sentences. The second

[1] `http://www.cs.umd.edu/~mount/ANN/`

Set	Training	Dev	Test	Vocab
RM	3.8	0.27	0.83	1000
SWB-100	3.79	1.27	1.26	100

Table 1. Vocabulary sizes and data set sizes (in hrs) for Resource Management (RM) and Switchboard subset (SWB-100) tasks.

task is a subset of the Switchboard-I dataset that was selected to have a limited vocabulary while being acoustically rich. Rather than using ad-hoc heuristics [9] to select this subset, we use a method similar to [10], which is based on submodular function optimization. Briefly, we select data to maximize be maximally diverse acoustically (as indicated by the number of different phonetic states represented in the subset) while limiting the number of different words in the subset. Selection is done using a greedy algorithm. This set, which we refer to as the SWB-100 dataset, is split into three parts for training, development and testing, respectively. These tasks were chosen because they are small vocabulary tasks that nevertheless represent a fair amount of acoustic variation by virtue of having multiple speakers and recording conditions. Compared to a large-vocabulary task, less effort needs to be spent on language modeling, which allows us to concentrate on our goal of testing novel acoustic modeling techniques. Table 1 shows the sizes of the data sets (amount of non-silence speech) and of the vocabularies. All speech data was preprocessed by extracting 13-dimensional feature vectors (consisting of 12 MFCC coefficients, 1 energy coefficient) every 10ms with a window of 25ms. Cepstral means are substracted on a per speaker basis; delta and delta-delta features are appended, resulting in 39-dimensional MFCC features. We also concatenate feature vectors across 9 adjacent frames and use LDA to project the features down to a 40-dimensional feature vectors.

4.1. Recognition systems

We use the Kaldi toolkit [11] to train all systems. First, a monophone GMM system is trained a small subset of the training data, which is then expanded into a triphone GMM/HMM system. The alignments from this system are used to bootstrap a triphone DNN/HMM system. There are currently two versions of DNN training recipes in Kaldi – the first implementation is based on [12], in which standard Restricted Boltzmann Machines (RBM), pre-training and stochastic gradient descent (SGD) training are used with GPUs. We adopt the second version of the DNN training recipes, which supports parallel training on multiple CPU and uses greedy layer-wise supervised training or layer-wise backpropagation [13, 14]. For detailed information on the training procedure readers are referred to [15]. For each task in our experiments we create two DNNs, shown in Figure 1. Network 1 includes a bottleneck layer between the last two

Task	Network 1					
	I	H1	H2	B	H3	O
RM	40	1024	1024	42	1024	1421
SWB-100	40	1185	1185	40	1185	624
	Network 2					
	I	H1	H2	H3	H4	O
RM	40	1070	1070	1070	1070	1421
SWB-100	40	1185	1185	1185	1185	624

Table 2. Sizes of networks' input (I), hidden (H), bottleneck (B) and output (O) layers.

hidden layers. Its linear outputs are used in the subsequent step of graph construction (see Section 3). Network 2 is a 4-layer DNN without a bottleneck layer – this is constitutes our baseline system as it typically achieves a better performance than a network with a bottleneck layer (e.g., the difference in WER on the SWB-100 dev set is 29.14% (Network 1) vs. 27.40% (Network 2)). The input layers of all networks consist of the LDA-transformed spliced MFCC features; the number of nodes in the output layers equals the number of HMM states used in the systems. The sizes of the networks' layers are listed in Table 2. The state priors needed for

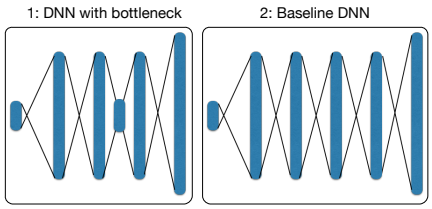

Fig. 1. DNN architectures.

conversion of posteriors to likelihoods are computed from the state occupancy counts in a forced alignment of the training data. The language models used in the systems are trigram models for the SWB-100 task and bigram models for the RM task. They were trained using SRILM with modified Kneser-Ney smoothing.

5. EXPERIMENTS AND RESULTS

We evaluate our method using WER as well as the accuracy of the HMM state probability predictions output by either the DNN or the GBL. To compute the latter, the index of the HMM state with the highest posterior is taken as the predicted label for each frame. Since we do not have manual annotations of the HMM state sequences for the test data, reference labels are obtained via forced alignment of the test transcriptions to the acoustic data using the recognition system with Network 2. Silence frames are excluded since they are always detected very accurately.

5.1. SWB-100 experiments

Our initial experiments focused on identifying the best feature representation for computing the pairwise similarity measure in Equation 2. In [2, 3, 4], windows of MFCC or PLP feature vectors were used, and Euclidean distance or cosine distance were used to derive the similarity measure. In [1], a multilayer perceptron was trained to convert the original MFCC features to a probability distributions over the desired classes. Jensen-Shannon divergence was then used to produce a similarity measure. In [5] these approaches were compared under the same conditions, and the latter was shown to perform better.

Comparison of different feature representations. Here we compare four different types of feature representations that can be directly extracted from the trained DNN. **1) LDA features:** The LDA-processed acoustic feature vectors (see Section 4) are used directly. **2) DNN posterior features:** These are the DNN-generated state posterior distributions, with a window of 2 frames to either side of the current frame (dimension: $624 \times 5 = 1320$); **3) DNN hidden-layer features (HLF):** These are the linear outputs from the last hidden layer in the DNN; **4) DNN Bottleneck features (BNF):** The bottleneck layer outputs from the Network 1 architecture (see Figure 1) are used with a window of 9 adjacent frames per sample (dimension: $40 \times 9 = 360$). Computation of the similarity measure (Equation 2) and graph construction were identical for each condition; each node was limited to its 10 nearest neighbors. The values for μ and ν in Equation 1 were 10^{-6} and 8×10^{-6}, respectively.

Table 3 shows a comparison of the resulting frame-level HMM state classification accuracies and WERs on the SWB-100 dev set. We see that the the state classification accuracy improves significantly ($\rho = 0.001$, using a difference of proportions significance test) in all GBL systems. The bottleneck features yield the best performance, outperforming the baseline DNN by a relative improvement of over 20%. Better frame-level accuracy does not always result in lower WER; however, the best GBL system achieves an absolute WER reduction of about 1%.

The comparison of the different feature representations is fairly self-explanatory: The raw MFCC+LDA features, which form the input layer to our networks, may be too noisy to discriminate between different context-dependent phone states. Posterior features, on the other hand, typically represent sharply peaked, low-entropy distributions that tend to introduce search errors during the subsequent Viterbi decoding process when predictions are wrong but highly confident. Hidden layer features and the bottleneck features are more discriminative than the input features but at the same time more fine-grained than the posterior features. The bottleneck feature outperforms the hidden layer feature because, due to their low dimensionality, they are concatenated across 4 frames, thus providing context-dependent information. Similar splicing could be applied to hidden layer features but the resulting high dimensionality of the feature vectors would slow down the graph construction process.

Speaker-dependent vs. speaker-independent graph construction. We also compared speaker-dependent with speaker-independent graph construction. Recall that the underlying assumption of our approach is that the test data lies on a manifold. The manifold structure is dependent on the characteristics of the test data, such as speaker, channel, and noise. Constructing a graph separately for each speaker rather than globally for the entire test data might therefore be a better choice. On the other hand, speaker-independent graph construction integrates larger unlabeled data sets into the learning process. Table 4 shows the results on the SWB-100 dev set; speaker-dependent graph construction yields better results. The linear score combination weights were also optimized on the SWB-100 dev set and were 1, 0.3, and 20 for the GBL score, baseline acoustic model score, and LM score, respectively.

System	Frame Acc.	WER
DNN Baseline	36.37%	27.40%
GBL: LDA features	**37.33%**	27.06%
GBL: DNN posterior features	**41.10%**	28.42%
GBL: DNN hidden-layer features	**41.53%**	26.99%
GBL: DNN bottleneck features	**43.85%**	26.45%

Table 3. Frame-level HMM state prediction accuracy (Frame Acc.) and WER for different feature representations, SWB-100 dev set. Boldface numbers are statistically significant improvements at $\rho = 0.001$.

SI graph		SD graph	
Acc	WER	Acc	WER
42.80%	27.65	43.85%	26.45

Table 4. Frame-level HMM state prediction accuracy and WER for speaker-independent (SI) vs.speaker-dependent (SD) graphs on the SWB-100 dev set.

System	Trigram	Unigram
DNN	29.11%	40.64%
GBL	28.69%	39.06%

Table 5. WERs on SWB-100 test set for baseline and best GBL system.

Table 5 shows results obtained on the SWB-100 test set, using the best setup identified above (score weights: 1, 0.3, and 20). In addition to using a standard trigram for decoding, we also ran a decoding pass with a unigram language model

RM	Test Feb89 (Dev)		Test Feb91		Test Oct89		Test Sep92	
	DNN	GBL	DNN	GBL	DNN	GBL	DNN	GBL
Frame Acc.	58.91%	**64.44%**	57.91%	**63.09%**	58.49%	**63.98%**	54.51%	**60.19%**
WER	2.42%	2.26%	1.73%	1.73%	2.68%	2.46%	4.03%	3.52%

Table 6. Frame accuracy and WER on Resource Management datasets.

(score weights: 1, 0.5, and 15) to better assess the contribution of our method under a weak language model. We see a consistent improvement over the DNN/HMM system in both conditions.

5.2. RM experiments

Table 6 shows frame-level accuracy and word error rates on our second task, the Resource Management corpus. For these experiments we used the same setup as before (i.e., bottleneck features and speaker-dependent graph construction). The linear score combination weights were re-optimized for this task and were 1 for both the baseline acoustic model and the GBL score, and 6 for the LM score. The baseline system is a state-of-the-art DNN system with a bigram LM, which already achieves a very low word error rate on this task. However, the GBL system gains a further improvement on all test sets except for *test-feb91*, where the word error rate remains constant.

5.3. Comparison with self-training

Finally, we compare our results against self-training. During self-training, an already-trained recognizer is applied to novel, untranscribed data. The unlabeled samples receive the highest likelihood or confidence during decoding are then selected and added to the original training set, along with the hypothesized transcription. The entire system is then re-trained. This procedure can be iterated several times. This is a different way of utilizing unsupervised data that has shown benefits for DNN/HMM systems (e.g.,[16, 17]). We ran the trained DNN/HMM system on the test data and selected the test utterances with the highest per-frame decoding likelihood. Selection was stopped when the number of frames in the selected set was equal to the number of test samples used by the GBL systems, to ensure comparable conditions. This resulted in about 1100 utterances for SWB and 260 utterances for RM). Three iterations of self-training were performed, after which no further improvements were obtained. Results are shown in Table 7. Self-training achieves better results on SWB-100 and one of the RM test sets while GBL yields better results on the remaining two RM test sets. Differences are not significant; however, GBL can be done in a single iteration and is very fast in itself. Graph construction and inference for the graphs used in this study took less than 10 minutes and 1-2 minutes, respectively on a 64-core Intel 2.00GHz machine. By contrast, one iteration of baseline system retraining took 1 hour (with DNN training parallelized across 64 CPUs). Thus, while the overall performance of GBL is similar to that of self-training, GBL is compuationally more efficient.

6. RELATED WORK

Several previous papers have addressed the topic of semi-supervised learning in speech recognition. Unlabeled data was used to train GMMs using a maximum mutual information criterion in [18]. Most approaches fall into the category of self-training, as described in the previous section. Originally, self-training was used in the context of GMM/HMM based sytems [19, 20]. More recently, it has been applied to DNN/HMM systems [16, 21, 22, 17]. Another class of approaches utilizes untranscribed data at the DNN pre-training stage [23, 17]. Since pre-training is an unsupervised procedure, no labels are necessary, and the test data can simply be added to the original training data. Our approach is different from of these in that we utilize a learning algorithm that is trained *jointly* on both the training and the test set in a single pass (i.e., non-iteratively). Most importantly, it explicitly exploits *similarities between different test samples* as well as similarities between training and test samples to bias the decision function, thus contributing complementary information. All other methods mentioned above process each test sample in isolation and do not model dependencies between different test samples. Initial studies on GBL-based acoustic modeling were reported in [1, 2, 3, 4, 5]; however, all of these works have focused on phonetic classification only. This paper is the first that takes GBL beyond phone classification and studies its effects on word recognition. Finally, [24] proposed an objective function for DNNs that contains a graph-based semi-supervised term as a regularizer; however, this was not tested on speech data. In [25] a similar criterion was added to a (non-deep) multi-layer perceptron, which achieved improvements in vowel classification.

7. CONCLUSIONS

We have presented an acoustic modeling technique that combines graph-based semi-supervised learning with DNN/HMM based speech recognition. GBL is used to improve the accuracy of the HMM state prediction at the frame-level; the resulting scores are integrated into the word hypothesis lattice before re-scoring. This technique achieves a statistically significant improvement in state prediction accuracy and a con-

	RM				SWB-100	
System	Test Feb89 (Dev)	Test Feb91	Test Oct89	Test Sep92	dev	test
Baseline	2.42%	1.73%	2.68%	4.03%	27.40%	29.11%
Self-training	1.99%	**1.69%**	2.98%	3.67%	26.45%	**28.01%**
GBL	2.26%	1.73%	**2.46%**	**3.52%**	26.45%	28.69%

Table 7. WERs for baseline system, self-trained system, and GBL. Boldface numbers indicate the best-performing system.

sistent reduction in word error rate. Moreover, it slightly outperforms self-training on one of the tasks, with much less time required than iterative self-training. Our future goal is to extend the current framework to a large-vocabulary recognition task. Moreover, GBL could be combined with self-training in that the more accurate hypotheses produced by GBL on the unlabeled data could be used in an active-learning framework, i.e., GBL and self-training could be interleaved. Finally, we are going to apply GBL to graphs constructed at the word level rather than the frame level.

8. REFERENCES

[1] A. Alexandrescu and K. Kirchhoff, "Graph-based learning for phonetic classification," in *Proceedings of ASRU*, 2007.

[2] A. Subramanya and J. Bilmes, "Entropic graph regularization in non-parametric semi-supervised classification," in *Proceedings of NIPS*, December 2009.

[3] K. Kirchhoff and A. Alexandrescu, "Phonetic classification using controlled random walks," in *Proceedings of Interspeech*, 2011.

[4] M. Orbach and K. Crammer, "Transductive phoneme classification using local scaling and confidence," in *Proceedings of IEEE 27th Convention of Electric and Electronics Engineers in Israel*, 2012.

[5] Y. Liu and K. Kirchhoff, "Graph-based semi-supervised learning for phone and segment classification," in *Proceedings of Interspeech*, 2013.

[6] X. Zhu and Z. Ghahramani, "Learning from labeled and unlabeled data with label propagation," Tech. Rep., CMU-CALD-02, 2002.

[7] P.P. Talukdar and K. Crammer, "New regularized algorithms for transductive learning," in *Proceedings of ECML-PKDD*, 2009, pp. 442–457.

[8] Y. Liu and K. Kirchhoff, "A comparison of graph construction and learning algorithms for graph-based phonetic classification," *UWEE Technical Report, UWEETR-2012-0005*, 2012.

[9] S. King, "SVitchboard 1: Small vocabulary tasks from Switchboard 1," in *Proceedings of Interspeech*, 2005, pp. 3385–3388.

[10] K. Wei, Y. Liu, K. Kirchhoff, C. Bartels, and J. Bilmes, "Submodular subset selection for large-scale speech training data," *Proceedings of ICASSP*, 2014.

[11] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, et al., "The Kaldi speech recognition toolkit," in *Proceedings of ASRU*, 2011, pp. 1–4.

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

[13] Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al., "Greedy layer-wise training of deep networks," *Proceedings of NIPS*, vol. 19, pp. 153, 2007.

[14] F. Seide, G. Li, and D. Yu, "Conversational speech transcription using context-dependent deep neural networks.," in *Proceedings of Interspeech*, 2011, pp. 437–440.

[15] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," *Proceedings of ICASSP*, 2014.

[16] K. Vesely, M. Hannemann, and L. Burget, "Semi-supervised training of deep neural networks," in *Proceedings of ASRU*, 2013, pp. 267–272.

[17] D. Imseng, B. Potard, P. Motlicek, A. Nanchen, and H. Bourlard, "Exploiting un-transcribed foreign data for speech recognition in well-resourced languages," in *Proceedings of ICASSP*, 2014.

[18] J.T. Huang and M. Hasegawa-Johnson, "Maximum mutual information estimation with unlabeled data for phonetic classification," in *Proceedings of Interspeech*, 2008.

[19] T. Kemp and A. Waibel, "Unsupervised training of a speech recognizer: recent experiments," in *Eurospeech*, 1999.

[20] L. Lamel, J.L. Gauvain, and G. Adda, "Lightly supervised and unsupervised acoustic model training," *Computer, Speech and Language*, pp. 115–129, 2002.

[21] S. Thomas, M.L. Seltzer, K. Church, and H. Hermansky, "Deep neural network features and semi-supervised training for low resource speech recognition," in *Proceedings of ICASSP*, 2013, pp. 6704–6708.

[22] F. Grezl and M. Karafiát, "Semi-supervised bootstrapping approach for neural network feature extractor training," in *Proceedings of ASRU*, 2013, pp. 470–475.

[23] P. Swietojanski, A. Ghoshal, and S. Renals, "Unsupervised cross-lingual knowledge transfer in DNN-based LVCSR," in *Proceedings of SLT*, 2012, pp. 246–251.

[24] J. Weston, F. Ratke, H. Mobahi, and R. Collobert, "Deep learning via semi-supervised embedding," in *Proceedings of ICML*, 2008.

[25] J. Malkin, A. Subramanya, and J. Bilmes, "On the semi-supervised learning of multi-layered perceptrons.," in *Proceedings of Interspeech*, 2009, pp. 660–663.