Odyssey 2014:
The Speaker and Language Recognition Workshop
16-19 June 2014, Joensuu, Finland

Application of Convolutional Neural Networks to Language Identification in Noisy Conditions

Yun Lei, Luciana Ferrer, Aaron Lawson, Mitchell McLaren, Nicolas Scheffer

Speech Technology and Research Laboratory, SRI International, California, USA
{yunlei,lferrer,aaron,mitch,scheffer}@speech.sri.com

Abstract

This paper proposes two novel frontends for robust language identification (LID) using a convolutional neural network (CNN) trained for automatic speech recognition (ASR). In the CNN/i-vector frontend, the CNN is used to obtain the posterior probabilities for i-vector training and extraction instead of a universal background model (UBM). The CNN/posterior frontend is somewhat similar to a phonetic system in that the occupation counts of (tied) triphone states (senones) given by the CNN are used for classification. They are compressed to a low dimensional vector using probabilistic principal component analysis (PPCA). Evaluated on heavily degraded speech data, the proposed front ends provide significant improvements of up to 50% on average equal error rate compared to a UBM/i-vector baseline. Moreover, the proposed frontends are complementary and give significant gains of up to 20% relative to the best single system when combined.

1. Introduction

The i-vector framework, originally developed for speaker recognition [1], has been successfully used as a feature extraction frontend for language identification (LID) [2]. In speaker recognition, probabilistic linear discriminant analysis (PLDA) [3] is used to generate verification scores using the i-vectors as input. For LID, a Gaussian backend is commonly used to model the languages and generate the final scores. Recent work [4, 5], however, has shown that a neural network can outperform a Gaussian backend in noisy conditions.

Another family of LID approaches is based on the modeling of phonetic sequences produced by open-phone loop recognizers, such as parallel phone recognition and language modeling (PPRLM) [6, 7], and phoneme posteriogram [8]. In general, the phone-based approaches perform comparably to the i-vector-based approaches; the fusion of those two systems results in significant improvements (e.g., [8, 9]).

Although state-of-the-art LID systems achieve good performance in clean conditions, noisy conditions still pose a considerable challenge. In this study, we focus on noise-robust LID using data released by the Defense Advanced Research Projects Agency (DARPA) under the Robust Automatic Transcription of Speech (RATS) program. The RATS data is heavily degraded by time-varying channel distortions.

In a recent article [10], we proposed a new speaker verification framework in which we used a deep neural network (DNN) trained for automatic speech recognition (ASR) to generate the posterior probabilities for a set of states that replace the Gaussians in the traditional UBM-GMM approach. The new framework exhibited significant improvements on the clean and collected noise telephone conditions of the 2012 NIST speaker recognition evaluation data.

In this paper, we show how this new paradigm can be successfully applied to the LID task. Specifically, we demonstrate how our approach can generalize to the use of convolutional neural networks (CNN) instead of DNN. ASR studies demonstrated that the CNN modeling is more robust against noise distortions than DNN modeling [11], thus making it an ideal candidate for this task.

Finally, we propose a new approach where the state posterior counts from the CNN are used directly for language identification. This approach is inspired by both the phoneme posteriogram and PRLM approaches to language identification [8, 12]. Both of the approaches proposed in this work generate vectors of a fixed dimension that can be further modeled in a standard fashion.

In the following sections, we first introduce some important concepts needed to understand the proposed CNN/i-vector and CNN/posterior systems described in Sections 3 and 4. The experiments and results will then be presented before this paper concludes.

2. Posterior extraction using CNN

The proposed approaches detailed in the following sections rely on the posterior probability generated by a CNN for a set of ASR states called *senones*. In this section, we introduce the concept of senones and describe how their posteriors are modeled using CNNs.

2.1. Senones in ASR

The senones are defined as tied states within context-dependent phones. They can represent, for example, each one of the three states within all triphones. They are the unit for which observation probabilities are computed during ASR. Therefore, the pronunciations of all words are represented by a sequence of senones $\mathcal{Q}$. In general, the set of senones is automatically defined by a decision tree using a maximum likelihood (ML) approach [13]. The decision tree is grown by asking a set of locally optimal questions that give the largest likelihood increase, assuming that the data on each side of the split can be modeled by a single Gaussian. The leaves of the decision tree are then

This material is based on work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract D10PC20024. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA or its contracting agent, the U.S. Department of the Interior, National Business Center, Acquisition & Property Management Division, Southwest Branch. The views expressed are those of the authors and do not reflect the official policy or position of the Department of Defense or the U.S. Government. "A" (Approved for Public Release, Distribution Unlimited)

taken as the final set of senones.

Once the set of senones is defined, a Viterbi decoder is used to align the training data into the corresponding senones using a preexisting ASR model. These alignments are used to estimate the observation probability distribution $p(x|q)$, where x is an observation vector (the acoustic features) and q is the senone. The estimation of the observation probability distribution and the realignment can be optimized alternatively and iteratively.

2.2. CNN for speech recognition

Traditionally, a speech recognition system uses Gaussian mixture models (GMM) to model the likelihood for each senone $p(x|q)$. Recent studies have shown that DNNs are better at estimating the senone posteriors $p(q|x)$ than GMMs [14, 15]. The observation probability can then be obtained from the posteriors and priors of the senones using Bayes rule, as follows:

$$p(x|q) = p(q|x)p(x)/p(q), \quad (1)$$

where $p(x|q)$ is the observation probability needed for decoding, $p(q)$ is the senone prior and $p(q|x)$ is the senone posterior obtained from the DNN. Since $p(x)$ is not available, the scaled likelihood $p(x|q)/p(x)$ is normally used during ASR decoding.

For noisy conditions, CNNs were proposed to replace DNNs to improve robustness against frequency distortion. A CNN is a neural network where the first layer is composed of a convolutional filter followed by max-pooling. The rest of the layers are identical to those of a standard DNN. CNNs were first introduced for image processing by [16, 17], and later used for speech recognition [18, 19]. Typically, for speech, the input features given to a CNN are the log mel-filterbank coefficients. Figure 1 presents an example of a convolutional layer. The target frame is generally accompanied by context information that includes several filter bank feature vectors around the target frame. One or more convolutional filters are then applied to the feature matrix. While in image processing this filter is generally small with respect to the size of the input image, in ASR, the filter is defined with the same length as the total number of frames. In effect, this means that no convolution happens in the time domain: a single weighted sum is done across time. On the other hand, the filter is generally much shorter than the number of filter banks. This way, the output is a single vector whose components are obtained by doing a weighted sum of several rows of the input matrix. For speech recognition, it was found in [20] that convolution performed only on the filter bank axis performed as well as convolution on both axes.

The dimension of the output vector of the convolutional layer depends on the number of filter banks and the height of the convolutional filter. In Figure 1, there are 7-dimensional filter bank features from 5 frames (2 left, 2 right and 1 center frames) used to represent one center/target frame. The height of the convolutional filter is 2 and its width is, as mentioned above, equal to the number of frames included in the input. Since we do not extend the input to deal with boundary issues, the output of the convolution is a 6-dimensional vector.

After the convolutional filter is applied, the resulting vector goes through a process called max-pooling by which the maximum value from N adjacent elements is selected. This process can be done with or without overlap. The process of max-pooling is expected to add robustness to noisy and channel distortions by picking the largest value from a set of adjacent filter banks that have already gone through convolutional filtering. The result of this example, where the pooling size is 3 and no overlap is used, is a 2-dimensional output.

In practice, 40 filter banks with a context of 15 frames are used, and the height of the convolutional filter is 8. Many convolutional filters can be used to model the data in more detail; we used 200 in this work. The output vectors of the different filters are concatenated into a long vector that is then input to a traditional DNN. This DNN usually includes 5 to 7 hidden layers. The output layer of the DNN contains one node for each senone defined by the decision tree.

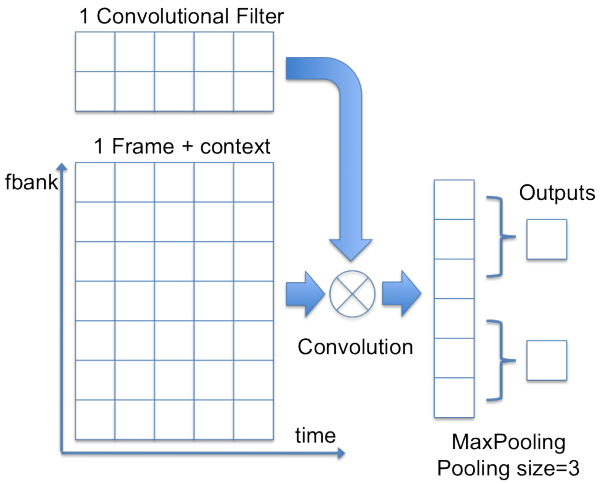

Figure 1: Diagram of a convolutional layer including convolution and max-pooling. Only one convolutional filter is shown in this example and non-overlapping pooling is used.

A flow diagram for CNN training in ASR is shown in Figure 2. A hidden Markov model (HMM) ASR system with GMM states is trained on 39 dimensional MFCC features and used to generate reliable alignments for the subsequent CNN training where 40 dimensional filter bank features are used. The final acoustic model is composed of the original HMM from the previous HMM-GMM system and the new CNN.

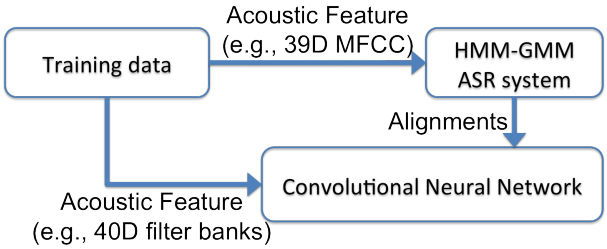

Figure 2: Flow diagram for CNN training for ASR.

3. CNN/i-vector System

The i-vector framework initially developed for speaker recognition gives excellent performance for LID [2]. This approach extracts a low-dimensional vector representing a waveform using the maximum likelihood criteria. A Gaussian backend, neural network, or other standard machine learning approaches can then be used to generate the likelihood scores for each language.

In state-of-the-art systems, a UBM is used to estimate the posterior probabilities for every frame to further generate the Baum-Welch statistics necessary for i-vector training and extraction. In [10], we proposed a new paradigm in which a DNN, trained for ASR, replaces the UBM to estimate the posterior probability for each frame (also known as frame alignment). From these posteriors, the resulting Baum-Welch statistics can be computed easily using a new set of features (such as MFCCs) The statistics are then whitened by the means and covariance of single Gaussians for each state estimated on a larget set of training data.

We demonstrated that this approach gives significant improvements in speaker recognition accuracy. This work aims at evaluating this new framework for the LID task. Additionally, a different type of network, CNN, is used in order to improve noise robustness, since the target data in this study is speech that is highly degraded by noise and channel distortions.

Figure 3 presents the flow diagram of the CNN/i-vector hybrid system for i-vector modeling. First, a CNN trained for ASR is used to extract the posteriors for every frame. Then, instead of the posteriors generated by the UBM in the traditional UBM/i-vector framework, the posteriors from the CNN are used to estimate the zeroth and first order statistics for the following i-vector model training. Note that an important characteristic of the approach is that one does not have to compromise by designing a feature that works well for both senone posterior estimation and i-vector estimation. Indeed, the first order statistics extraction in the i-vector model can use completely different features from the features used for the CNN. While the CNN features should contain information about the phonetic content, the features used for i-vector extraction should contain information about the spoken language. Both sets of features can be chosen separately to improve the final LID performance.

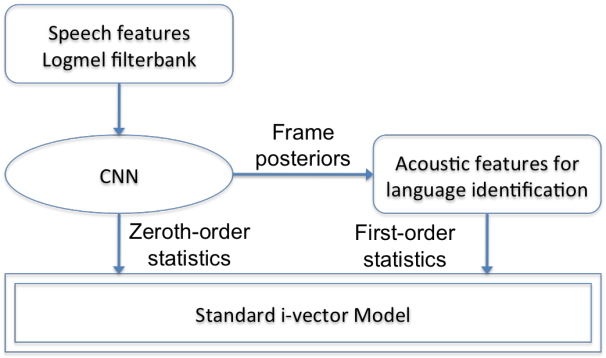

Figure 3: The flow diagram of CNN/i-vector hybrid system for i-vector model.

4. CNN/Posterior System

The CNN/i-vector approach described in the previous section can be considered part of the family of LID methods that focus on modeling short-time acoustic features. Another family of methods focus on modeling the sequence of phonetic units, given by a phone recognizer. Standard approaches involve collecting the probabilities and counts of phone sequences as a representation of the signal using the output of one or several open-phone loop recognizers [6, 7]. Another approach uses the phoneme posteriogram counts from the phone recognizer to create bigram conditional probabilities which are then used to create features for LID (e.g., [8]).

Inspired by these approaches, we propose to use the senone posteriors obtained by the CNN for LID. Since senones are defined as states coming from context-dependent phones, their posterior probabilities are intrinsically contextualized by the phonetic context. As such, we argue that our method alleviates the need of the standard approaches for contextualization which is done using Ngram modeling. Furthermore, we fully utilize the benefit of CNN modeling's relative robustness to noise, channel distortions and speaker variability. That is, we expect that the occupation counts given by the CNN capture the frequency of usage of each senone for the language present in a sample, and that they are relatively independent of other conditions found in the signal.

The counts for each senone q on the ith file are given by

$$Z_q(i) = \sum_{t|t\in S} \gamma_q(i,t),\ q \in Q \tag{2}$$

where $\gamma_q(i,t)$, the posterior of the senone q given the tth frame of the ith file, is obtained by the CNN, Q is the set of senones, and S corresponds to the set of speech frames as determined by a speech activity detector. Similarly to lattice counts used by PRLM approaches, this count does not depend on hard decisions from the recognizer. To create the log-posterior features, we divide these values by the number of speech frames N_S and take the logarithm.

$$\hat{W}_q(i) = \log Z_q(i)/N_S \tag{3}$$

Finally, we form a feature vector to represent waveform i by concatenating these values for all senones

$$\hat{W}(i) = \left[\hat{W}_0(i),\ \hat{W}_1(i),\ \cdots,\ \hat{W}_Q(i)\right]^T \tag{4}$$

The dimension of the resulting vector is equal to the number of senones (usually between 3K and 7K), which is much larger than the usual size of the i-vector modeled by standard backend approaches for LID. We use probabilistic principal component analysis (PPCA) to reduce the dimension of $\hat{W}(i)$, similarly to what was done in [21] for MLLR features. Mean and variance normalization is applied prior to PPCA. These vectors can then be modeled with standard backends used for i-vectors.

In contrast to the i-vector based approaches in the previous section, where the zeroth and first order statistics are used, the proposed CNN/posterior approach only uses zeroth order statistics to estimate the low-dimensional vector. The information available to the backend from these two systems are thus very different, even though the process at the core of these two systems, the CNN, is identical. We will show further that these two approaches are complementary and give a significant improvement when used in combination.

5. Experiments

In this study, we evaluated the proposed approaches on the RATS LID task consisting of five target languages (Farsi, Urdu, Pashto, Arabic Levantine and Dari) and a pre-defined set of out-of-set languages [5, 22]. Clean conversational telephone recordings were retransmitted over seven channels for the RATS program (the eighth channel D was excluded from the LID task). The signal-to-noise ratio (SNR) of retransmitted signals ranged between 30dB to 0dB. Four conditions are considered in which test signals are constrained to have duration close to 3, 10, 30 and 120 seconds. The details of the task can be found in [23].

5.1. System setup

The baseline system used in this study is the standard UBM/i-vector system followed by a neural network backend. The data used for training the UBM and i-vector models include five target languages and other out-of-set languages from the RATS LID training set. Similar to [24], a 140 dimensional 2D-DCT feature optimized for RATS LID task are used for the UBM/i-vector framework. A 2048 diagonal component UBM is trained in a gender-independent fashion, along with a 400 dimensional i-vector extractor.

To extract the posterior probability of the senones, both HMM-GMM and HMM-CNN models are trained on the RATS KWS training data which only contains Arabic Levantine and Farsi. The following experiments confirm that the posterior from the CNN can result in good performance despite the fact that only two languages were used for training. The 3353 senones are generated by the decision tree and the cross-word triphone HMM-GMM ASR with 200k Gaussians are trained with maximum likelihood (ML). The features used in the HMM-GMM model are 39-dimensional MFCC features, including 13 static features (including C0) and first and second order derivatives. The features were pre-processed with speaker-based cepstral mean and covariance normalization (MVN).

A CNN was trained using cross entropy on the alignments from the HMM-GMM. The input features are given by 40 log mel-filterbank coefficients with a context of 7 frames from each side of the center frame for which predictions are made. Two hundred convolutional filters of size 8 are used in the convolutional layer and the pooling size is set to three without overlap. The subsequent DNN includes five hidden layers, with 1200 nodes per hidden layer, and the output layer, with 3353 nodes representing the senones.

Neural network backends with a single hidden layer of 200 nodes are used for all presented systems. All input vectors to the NN backend are of size 400 (for the baseline system, CNN/i-vector and CNN/posterior). The output layer is of size 6, including the 5 target languages and an "unknown language" category. To optimize the performance on all durations, the original dataset is chunked into 8 and 30-second segments with 50% overlap. The chunked data as well as the original data are used together for NN training. The scores generated by the NN are further calibrated through logistic regression using 2-fold cross-validation on the test data.

The performance was evaluated on three measurements, including the average equal error rate (EER) over all target languages, the miss rate where the false alarm rate is equal to 1% widely used in the RATS programs [23], and the Cavg defined in the NIST Language Recognition Evaluation (LRE) 2009 [25].

5.2. Results

To make a fair comparison between the proposed CNN/i-vector hybrid system and the UBM/i-vector baseline system, the same 2D-DCT features are used in the CNN/i-vector system to compute the first-order sufficient statistics using the frame posteriors given by the CNN. In addition, as the CNN/i-vector system effectively use 3353 classes, another baseline system is presented where a supervised UBM is used to replace the standard UBM, similarly to what was done in [10]. Each Gaussian in the supervised UBM is trained using the frames assigned to one senone by the CNN posterior. This UBM is then used in the standard way to generate i-vectors.

The performance of the CNN/i-vector, CNN/posterior systems, the baseline systems and two fusion systems using linear logistic regression and 2-fold cross-validation are presented in Figure 4 for the three different error metrics. We can see that both the new CNN/i-vector and CNN/posterior systems achieve a significant improvement compared to the baseline systems across all duration conditions and error metrics. Furthermore, we can see that the fusion of the two proposed systems (called fusion 2-way in the figure) gives a significant improvement over both individual systems, indicating that these two systems are highly complementary. Finally, the addition of the UBM/i-vector system does not lead to consistent improvements. It might be caused by the large performance gap between the UBM/i-vector system and proposed systems. Table 1 shows the average EER numbers plotted in the top of Figure 4.

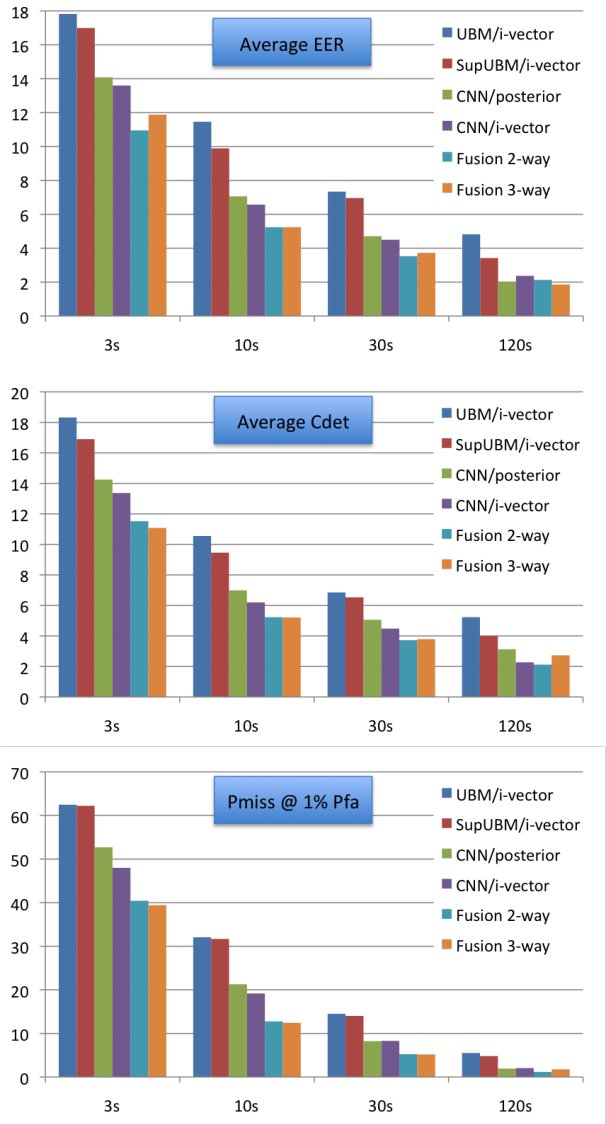

Figure 4: Average EER, average Cdet and Pmiss at 5% Pfa for different systems over the four duration conditions. Fusion 2-way refers to the fusion of the CNN/i-vector and CNN/posterior systems. Fusion 3-way refers to the fusion of those two systems as well as the UBM/i-vector system.

Table 1: *Average EER for different systems over the four duration conditions. Fusion 2-way refers to the fusion of the CNN/i-vector and CNN/posterior systems. Fusion 3-way refers to the fusion of those two systems as well as the UBM/i-vector system.*

System	3 sec	10 sec	30 sec	120 sec
UBM/i-vector	17.82	11.46	7.34	4.82
SupUBM/i-vector	16.99	9.89	6.96	3.42
CNN/posterior	14.08	7.06	4.71	2.00
CNN/i-vector	13.60	6.57	4.50	2.37
Fusion 2-way	10.95	5.24	3.53	2.13
Fusion 3-way	11.88	5.24	3.73	1.86

6. Conclusions

We introduced two novel front-ends for language identification that make use of the posterior probabilities generated by a CNN. The CNN was trained for prediction of a set of senones — the states within context-dependent phones — for speech recognition. The first system, called CNN/ivector system, uses the posterior probabilities as input to the standard i-vector training and extraction procedures. The second system, called CNN/posterior system, uses the posteriors to directly estimate a set of normalized zeroth order statistics for the senones. This vector is reduced in dimension and modeled using a standard neural network backend. While the second system only models the posteriors for the senones, the first system models the distribution of a set of selected features for each of the senones.

Since senones are constrained by phonetic context, the proposed systems implicitly model phonetic sequence information without the bigram or trigram modeling required by other phone-based approaches to language identification. Furthermore, since CNN posteriors are relatively robust to noise and channel degradations, the proposed front-ends are also adequate for this kind of challenging data.

Results of the newly proposed approaches were presented on heavily degraded speech data from the RATS LID task. We showed relative improvements between 23% and 50% on average EER with respect to a state-of-the-art UBM/i-vector system across different duration conditions. We also observed that the fusion of both approaches gives to a 20% relative performance gain over the best individual system, an improvement that implies these systems contain complementary information.

As next step, we plan to study the performance influence of a series of variables of the CNN for LID, such as the configurations, data languages, and evaluate the proposed approaches on more languages (e.g., NIST LRE tasks).

7. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. ASLP*, vol. 19, pp. 788–798, May 2010.

[2] M. Penagarikano, A. Varona, M. Diez, L. J. Rodriguez-Fuentes, and G. Bordel, "Study of different backends in a state-of-the-art language recognition system," in *Interspeech-2012*, 2012, pp. 2049–2052.

[3] S.J.D. Prince, "Probabilistic linear discriminant analysis for inferences about identity," in *ICCV-11th*. IEEE, 2007, pp. 1–8.

[4] M. McLaren, A. Lawson, Y. Lei, and N. Scheffer, "Adaptive gaussian backend for robust language identification," in *Interspeech-2013*, 2013, pp. 84–88.

[5] A. Lawson, M. McLaren, Y. Lei, V. Mitra, N. Scheffer, L. Ferrer, and M. Graciarena, "Improving language identification robustness to highly channel-degraded speech through multiple system fusion," in *Interspeech-2013*, 2013, pp. 1507–1510.

[6] P. Matejka, P. Schwarz, J. Cernocky, and P. Chytil, "Phonotactic language identification using high quality phoneme recognition," in *Interspeech-2005*, 2005.

[7] W. Shen, W. Campbell, T. Gleason, D. Reynolds, and E. Singer, "Experiments with lattice-based PPRLM language identification," in *Odyssey 2006 -The Speaker and Language Recognition Workshop*, 2006, pp. 1–6.

[8] L. F. DHaro, O. Glembek, O. Plchot, P. Matejka, M. Soufifar, R. Cordoba, and J. Cernocky, "Phonotactic language recognition using i-vectors and phoneme posteriogram counts," in *Interspeech-2012*, 2012, pp. 42–45.

[9] N. Brummer, S. Cumani, O. Glembek, M. Karafiat, P. Matejka, J. Pesan, O. Plchot, M. Soufifar, E. Villiers, and H. Cernocky, "Description and analysis of the brno276 system for LRE2011," in *Proceedings of Odyssey 2012: The Speaker and Language Recognition Workshop*, 2012, pp. 216–223.

[10] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *ICASSP-2014*, 2014.

[11] H. Soltau, H. Kuo, L. Mangu, G. Saon, and T. Beran, "Neural network acoustic models for the DARPA RATS program," in *Interspeech-2013*, 2013, pp. 3092–3096.

[12] Z.A. Zissman, "Comparison of four approaches to automatic language identification of telephone spech," *IEEE Trans. SAP*, vol. 4, pp. 31–44, Jan. 1996.

[13] S. J. Young, J. J. Odell, and P. C. Woodland, "Tree-based state tying for high accuracy acoustic modelling," in *HLT '94 Proceedings of the workshop on Human Language Technology*, 1994, pp. 307–312.

[14] G. Hinton, L. Deng, D. Yu, G.E. Dahl, A. Mohamed, N. Jaitly, A. Senior, V. Vanhoucke, P. Nguyen, T.N. Sainath, and B. Kingsbury, "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," *Signal Processing Magazine, IEEE*, vol. 29, no. 6, pp. 82–97, 2012.

[15] G.E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition," *IEEE Trans. ASLP*, vol. 20, pp. 30–42, 2012.

[16] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," *MIT Press*, pp. 255–258, 1995.

[17] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient based learning applied to document recognition," in *Proceedings of the IEEE*, 1998, pp. 2278 – 2324.

[18] O. Abdel-Hamidy, A. Mohamedz, H. Jiangy, and G. Penn, "Applying convolutional neural networks concepts to hybrid NN-HMM model for speech recognition," in *ICASSP-2012*, 2012, pp. 4277 – 4280.

[19] T. Sainath, A. Mohamed, B. Kingsbury, and B. Ramabhadran, "Deep convolutional neural networks for LVCSR," in *ICASSP-2013*, 2013, pp. 8614 – 8618.

[20] O. Abdel-Hamid, L. Deng, and D. Yu, "Exploring convolutional neural network structures and optimization techniques for speech recognition," in *Interspeech-2013*, 2013, pp. 3366–3370.

[21] N. Scheffer, Y. Lei, and L. Ferrer, "Factor analysis back ends for MLLR transforms in speaker recognition," in *Interspeech-2011*, 2011, pp. 257–260.

[22] K. Walker and S. Strassel, "The RATS radio traffic collection system," in *Odyssey 2012: The Speaker and Language Recognition Workshop*, 2012.

[23] "DARPA RATS program," http://www.darpa.mil/Our_Work/I2O/Programs/Robust_Automatic_Transcription_of_Speech_(RATS).aspx.

[24] M. Mclaren, N. Scheffer, L. Ferrer, and Y. Lei, "Effective use of DCTs for contextualizing features for speaker recognition," in *ICASSP-2014*, 2014.

[25] "National institute of standards and technology, the 2009 NIST language recognition evaluation plan," http://www.itl.nist.gov/iad/mig/tests/lre/2009/.