Augmented Data Training of Joint Acoustic/Phonotactic DNN i-vectors for NIST LRE15

Alan McCree, Gregory Sell, and Daniel Garcia-Romero

Human Language Technology Center of Excellence
Johns Hopkins University, Baltimore, MD, USA
alan.mccree@jhu.edu, gsell@jhu.edu, dgromero@jhu.edu

Abstract

This paper presents the JHU HLTCOE submission to the NIST 2015 Language Recognition Evaluation, including critical and novel algorithmic components, use of limited and augmented training data, and additional post-evaluation analysis and improvements. All of our systems used i-vectors based on Deep Neural Networks (DNNs) with discriminatively-trained Gaussian classifiers, and linear fusion was performed with duration-dependent scaling. A key innovation was the use of three different kinds of i-vectors: acoustic, phonotactic, and joint. In addition, data augmentation was used to overcome the limited training data of this evaluation. Post-evaluation analysis shows the benefits of these design decisions as well as further potential improvements.

1. Introduction

The 2015 Language Recognition Evaluation (LRE15) continues a long tradition of NIST conducting formal evaluations of language recognition technology to foster research progress in the field. Like previous evaluations in 2009 and 2011, this one used both conversational telephone speech and broadcast narrowband speech data, focused on confusable languages, and used a detection metric [1]. Unlike previous LREs, however, LRE15 introduced a limited training condition, where only specified speech material could be used to develop systems, as the core requirement. In addition, test segments were selected to cover a broad range of durations as opposed to the earlier focus specifically on 3, 10, and 30 seconds.

LRE15 used twenty languages grouped into six confusable language clusters: Arabic (Egyptian, Iraqi, Levantine, Maghrebi, Modern Standard), Chinese (Cantonese, Mandarin, Min, Wu), English (British, General American, Indian), French (West African, Haitian Creole), Slavic (Polish, Russian), and Iberian (Caribbean Spanish, European Spanish, Latin American Spanish, Brazilian Portuguese). The official performance metric C_{avg} was based on average Bayes cost as in previous LREs, but this time it was computed separately for each cluster and then averaged.

Most current research in language recognition revolves around i-vectors [2], which have provided the best performance in recent NIST evaluations of both speaker and language recognition. In this approach, built on techniques originally developed for subspace modeling of Gaussian Mixture Models (GMMs) using Joint Factor Analysis, the GMM for each speech cut is assumed to differ from a Universal Background Model (UBM) by only a low-dimensional offset of the mean supervector. The Maximum a Posteriori (MAP) estimate of this offset, called an *i-vector*, is generated for each cut and treated as an input feature for classification.

Recent research progress has used Deep Neural Network (DNNs) trained as acoustic models for speech recognition to improve modeling power in one of two ways: using bottleneck (BN) features from DNNs as inputs to a GMM/UBM [3, 4] or replacing the GMM/UBM frame alignment process with a supervised frame alignment into clustered phone states (senones) [5, 6, 7]. For these DNN approaches, language recognition systems also use counts of phone state posteriors to capture phonetic content in a *phonotactic* approach [8]. For this LRE, we found good performance with our recently developed approach of using multinomial i-vectors based on counts [9]. In addition, we propose a new and effective way to combine both acoustic and phonotactic information with a single *joint* i-vector extraction.

Data augmentation is the process of distorting, manipulating, or processing existing data in order to synthetically increase the amount or variety of the training data. This approach is especially meaningful in the case of labeled data where the annotations can be mapped to the perturbed data, thus creating additional data for supervised training without any additional costs. Within audio processing, augmentation strategies have shown value in recent years for automatic speech recognition [10, 11, 12, 13], and the limited training condition in LRE15 creates the opportunity for gains from augmentation for language recognition as well.

This paper describes our submission to LRE15, including system design, novel algorithmic components, use of limited and augmented training data, and additional post-evaluation analysis and improvements. The layout of this paper is as follows. In Section 2, we discuss the i-vectors used in the JHU HLTCOE submissions; then, in Section 3, we detail the specifics of the submission systems. Finally, Section 4, discusses post-evaluation analyses and improvements to the classifier and augmentations.

2. Acoustic, Phonotactic, and Joint I-vectors

For both DNN approaches, BN features and senone GMMs, we have extended the traditional acoustic i-vector model to capture the phonotactic information in the GMM frame counts [9]. Most recently, we have developed a new simple but effective joint i-vector extractor that simultaneously models both acoustic and phonotactic information with a single vector.

2.1. Acoustic i-vectors

In an acoustic i-vector system [2], each audio cut is assumed to have been generated by a GMM which only differs from the UBM by a linear subspace offset:

$$\mathbf{m}_i = \mathbf{m}_0 + \mathbf{T}_m \mathbf{z}_i \tag{1}$$

where $\mathbf{m}_i$ is a Gaussian supervector representing the stacked GMM means for cut i, the matrix $\mathbf{T}_m$ represents the total variability subspace over all such GMM models, and the subspace offset for cut i is $\mathbf{z}_i$. If the GMM alignments are given by the UBM and the subspace vector $\mathbf{z}_i$ has a standard normal prior, then the MAP estimate of the subspace offset is referred to as an i-vector. The matrix $\mathbf{T}_m$ is estimated using maximum likelihood (ML) over a large set of speech audio cuts with an EM algorithm.

2.2. Phonotactic i-vectors

Phonotactic information is captured by counts of phone usage, and can be modeled by a multinomial i-vector [14]. In this model, the observed counts are assumed to have been generated by a multinomial model $\mathbf{w}_i$ whose parameters depend on the speech file, and this model differs from a background model $\mathbf{w}_0$ by a linear offset in log probability space:

$$\mathbf{w}_i = \text{softmax}\left(\log(\mathbf{w}_0) + \mathbf{T}_w \mathbf{z}_i\right) \tag{2}$$

where the softmax operation both exponentiates from the log space back to linear and renormalizes to give a correct multinomial probability. This multinomial subspace model was first introduced in the subspace GMM approach to speech recognition [15]. The maximum likelihood (ML) estimate of this subspace offset is called a multinomial i-vector. Since there is no closed-form solution to this non-linear problem, an iterative algorithm based on Newton's method with some convergence heuristics is used.

Following [9], we generate multinomial i-vectors from GMM mixture usage patterns by summing frame posteriors instead of phone counts. While this can be done with unsupervised GMMs, we find better performance with these DNN systems since then count statistics capture phone state information instead of just acoustic similarity. Also, we continue to use the MAP estimate version of the algorithm rather than ML of [15], assuming a standard normal prior. We find that this improves performance for short duration files, and also helps convergence of the EM-like iterative $\mathbf{T}_w$ matrix estimation.

2.3. Joint i-vectors

Since the acoustic and phonotactic i-vectors focus on different aspects of the speech signal, it can be expected that combining them will improve performance. We have found gains with both score and i-vector fusion, but here we propose a more elegant alternative: to extract a single subspace offset that models both characteristics. This joint i-vector allows the GMM for cut i to differ from the UBM in both means and weights:

$$\mathbf{m}_i = \mathbf{m}_0 + \mathbf{T}_m \mathbf{z}_i \tag{3}$$

$$\mathbf{w}_i = \text{softmax}\left(\log(\mathbf{w}_0) + \mathbf{T}_w \mathbf{z}_i\right) \tag{4}$$

This can be viewed as a modified form of the subspace GMM, and this i-vector can be extracted using Newton's method with updates similar to [15]. To reduce complexity and improve convergence, we have introduced the following extensions to the joint i-vector extraction algorithm. First, we use MAP estimation, rather than ML, for more robust estimation with short durations and improved numerical stability. Second, we initialize the joint i-vector with the acoustic closed-form solution instead of a zero vector. Finally, we use a diagonal approximation to the Hessian matrix to greatly reduce the computation per iteration, utilizing an adaptive stepsize logic to ensure convergence, as in [9]. Without these modifications, 10 iterations of Newton's method require an order of magnitude increase in complexity of the i-vector extraction as compared to an acoustic i-vector; with them, the joint i-vector presents only a minor computation increase.

Joint estimation of the two matrices $\mathbf{T}_m$ and $\mathbf{T}_w$ is done with an EM-like algorithm. To improve convergence, we initialize $\mathbf{T}_m$ with a PCA analysis over a subset of the training data and start with $\mathbf{T}_w$ set to zero. We find similar convergence of this algorithm to the traditional acoustic case, and the additional computation to update $\mathbf{T}_w$ is small since this count matrix is much smaller than $\mathbf{T}_m$.

In all of our experiments, this joint i-vector provides at least a modest improvement over either acoustic or phonotactic alone. The biggest improvement comes for the senone case, which is the system examined for all three i-vectors in the post-evaluation analysis section below.

3. The JHU HLTCOE submission to LRE15

All systems in the JHU HLTCOE submissions used i-vectors based on DNNs. Classification was done with discriminatively-trained Gaussian classifiers [16]. In more detail, each system followed the same recipe:

- Speech activity detection with GMM SAD
- I-vector extraction
- Whitening and length normalization [17]
- Estimation of class means and shared within-class and across-class covariances
- Dimension reduction by Linear Discriminant Analysis (LDA)
- Refinement of parameters using discriminative training (MMI).

Individual systems differed in their use of DNNs (BN or senone) and types of i-vectors (acoustic, phonotactic, or joint). System combination was done with linear score fusion.

3.1. Training Data

The UBM/T training lists for the submission systems included the LRE15 training data as well as the included Switchboard and Switchboard Cellular files. Any file with more than 120 seconds of speech was segmented into sub-blocks with no more than 120 seconds each. This causal segmentation ideally would capitalize on speaker differences resulting from hand-offs (though would not account for conversational turn-taking) and increase the span of the training matrix. Apart from the segmentation, the audio was left untouched for the official submission UBM/T training.

Lists for training and tuning the language classification task (length-norm, LDA, MMI, and back-end) were generated from only the LRE15 training data. The files were randomly separated into an 80/20 split. Within each of these lists, the files were segmented into blocks containing between 3 and 30 seconds of speech. The length distribution was approximately uniform within this range, though, when possible, segmentation

followed speech activity to provide more natural boundaries. In the case of the lowest-resource languages, audio files were re-segmented to different lengths to increase the number of available trials.

Data was augmented via one of several distortions:

- **Resampling** - Sample rate perturbation, distorting pitch and speaking rate.
- **Additive Noise** - Multiband randomly-modulated Gaussian noise.
- **Reverberation** - Convolution with a synthetically generated impulse response.
- **Multiband Compression** - Dynamic range compression on eight linearly-spaced frequency bands.
- **Phone Channel Encoding** - GSM-AMR encoding at 6.7 or 4.75 kb/s.

For any given audio file, three of the segments were left unaugmented, while every other segment received a randomly selected augmentation. This strategy was repeated for both training and development lists.

3.2. DNN training

We trained two DNNs using the Kaldi speech recognition toolkit. One DNN was trained to compute senone posteriors and the other one to extract BN features. The labels (i.e. frame alignments to senones) for the DNNs are obtained from a standard tied-state triphone GMM-HMM system trained with maximum likelihood on the Switchboard 1 material provided by NIST. The senone set is obtained by clustering the states using a decision tree and the number of total Gaussians was set to 200K. The number of senones after the clustering was 9184. The input features for the GMM-HMM system are 40 dimensional vectors obtained from an LDA+MLLT projection of 7 spliced frames of 13 MFCCs. These features are further processed by an fMLLR transform to perform speaker adaptation.

Our DNN architecture uses 5 hidden layers of p-norm non-linearities with $p = 2$ and an input/output dimension of 5000/500 [18] for a total of 16M parameters. The input to the DNNs are 9 spliced vectors of the 40 dimensional LDA+MLLT projected features (total of 360 dim). fMLLR transforms are not used since they add significant computational complexity.

The DNN training algorithm performs back propagation using mini-batch pre-conditioned gradient descent and parameter averaging [19]. Data parallelization is accomplished by training n replicas of the DNN ($n = 4$ for our system) independently on disjoint subsets of data and combining them periodically by averaging their parameters. Once a new updated DNN is obtained, it gets replicated and asynchronous training is performed again. We refer to each one of these stages of "replicate-train-merge" as an iteration. An epoch (i.e. a run over the entire training dataset) consists of a fixed number of these iterations. During an iteration, each replica does back propagation over 300,000 training examples (using mini-batches of size 512). We use an exponential decay schedule for the learning rate and minimize negative cross entropy for a fixed number of epochs. Convergence is monitored on a validation set in terms of cross entropy and frame accuracy. After convergence, the BN DNN is initialized using a SVD low-rank matrix approximation [20] with a 60 dimensional BN between the 4th and 5th nonlinear layers of the DNN. The same training approach is used on this new architecture until convergence on the validation set. To obtain the BN features we strip the layers after the BN and perform a forward pass on the data.

Table 1: Official performance (average C_{avg}) of submitted systems. Primary submission is in bold.

System	Submission
[0] BN, joint	19.94
[1] Senone, acoustic	-
[2] Senone, phonotactic	-
[3] Senone, joint	19.74
[0,3] Fusion	18.77
[0,1,2] Fusion	**18.54**

3.3. MMI Gaussian Classifier

Most state-of-the-art systems for i-vector language recognition use classifiers such as Gaussian models, logistic regression, or cosine scoring, followed by a multiclass back-end which provides significant performance improvement as well as producing calibrated probability outputs [21]. We have recently demonstrated success using only a single step: a Gaussian classifier discriminatively trained using Maximum Mutual Information (MMI) [16]. This two-covariance model requires three parameters: the class means, a shared within-class covariance, and an across-class covariance. Initial sample covariance estimates of these parameters are used for dimension reduction using a diagonalizing form of LDA. The parameters are then refined with two-stage MMI training of first a scale factor of the within-class covariance and then the class means.

3.4. Fusion and Scoring

Score fusion was implemented by learning a scale factor and duration model [22] for each indidvidual system, averaging the scaled scores, and learning an overall scale factor for the fused system. Learning optimized multiclass cross-entropy to encourage calibrated identification posteriors.

The primary JHU HLTCOE submission fused 3 systems: joint i-vectors on bottleneck features, acoustic i-vectors using senone-supervised statistics, and phonotactic i-vectors on senone counts. Alternative submissions were two individual systems of joint i-vectors on BNs and joint i-vectors on senone-supervised statistics, as well as the fusion of these two.

Final scores applied Bayes' rule for each cluster of the languages to generate ID scores over each, and mapped these scores back to detection log likelihood ratios. This approach optimizes the closed set detection performance for each subset, at the expense of degraded performance across clusters.

3.5. Processing Speed

Processing speed was measured on a single 3.3 GHz Intel CPU for the first 100 cuts on the training list. For the primary submission, the bottleneck system required 3.8 seconds per cut to estimate features and stats, and 0.8 sec to estimate joint i-vectors. The senone DNN system needed 6.9 sec to estimate features and stats, 1.8 sec for acoustic i-vectors, and 0.3 sec for phonotactic i-vectors. Therefore total submission computation took approximately 13.6 seconds per audio file.

3.6. Submitted System Performance

Official performance of the JHU HLTCOE submissions is shown in Table 1. Since the task proved to be more difficult than

Table 2: C_{avg} results for each of the systems under several conditions: the submission conditions, the classifier initialized with ML point estimates, and the two best performing augmentation list combinations (as determined from Table 4).

System	Submission	ML Approach	Full; random, AugV1	Full; random, AugV2
Acoustic Baseline	-	23.85	22.21	22.17
[0] BN, joint	19.94	19.24	18.35	18.50
[1] Senone, acoustic	-	20.25	19.37	19.21
[2] Senone, phonotactic	-	20.94	20.50	20.34
[3] Senone, joint	19.74	19.16	18.50	18.38
[0,3] Fusion	18.77	18.12	17.35	17.33
[0,1,2] Fusion	18.54	18.01	17.40	**17.32**

expected, these are quite good numbers. The primary submission was a three-way score fusion of joint BN, senone acoustic, and senone phonotactic i-vector systems. This fusion performed better than each of the two individual joint i-vector systems, and slightly better than the alternative two-way fusion of those. Partly because of the joint i-vector approach, our senone system was competitive with the bottleneck version and we were one of the only sites to see a significant fusion gain.

4. Post-Evaluation Analysis and Improvements

4.1. Classifier Design

In [16], three different variations of the Gaussian MMI classifier were presented and compared. In the most straightforward *known model* case, the initial parameters are estimated using ML point estimates. In the *unknown model case*, Bayesian estimation is used to model limited training data uncertainties. Scoring uses the Gaussian predictive distribution, with a MAP mean and larger covariance. This algorithm can be viewed as an alternative form of the PLDA speaker recognition algorithm. A third variation is Bayesian estimation with a heuristic *single-cut* scoring assumption; this version is widely used in speaker recognition.

We typically find that before MMI training the third of these options provides the best performance for language recognition, although after MMI they are all quite similar. Therefore we used this configuation in our submission. However, for LRE15 this turned out to be a poor decision. As shown in Table 2, our submitted single-cut Bayesian version is in fact consistently outperformed by the ML approach. We do not currently have a good hypothesis for why the LRE15 evaluation causes this effect.

4.2. Joint i-vector Performance

Results for the senone system with acoustic, phonotactic, and joint i-vectors are also shown in Table 2. The joint approach provides a gain over both the acoustic and phonotactic methods for all list/classifier combinations shown. In fact, the joint score is often roughly 1% lower in C_{avg} than the acoustic i-vector score, which itself is always better than the phonotactic score. These results show that this joint i-vector is a better approach to estimating a single i-vector from GMM statistics.

4.3. Training Data Usage Analysis

In order to better understand the effects of the data manipulations included in the official submission systems, we ran several variations of the duration segmentations and distortions.

Table 3: Available parameters for each augmentation type under the two strategies considered (V2 was selected for the submissions). Each was evaluated as a mixture, and the individual augmentations were also tested separately with V2 parameters.

	V1	V2
Resample	{0.90;0.95;1.05;1.10}	{0.95;1.05}
Noise	SNR {6;12;18}	SNR {12;18}
Reverb	-	{short;long}
Compress	4 levels	2 levels
Phone	{6.7kb/s;4.75kb/s}	{6.7kb/s;4.75kb/s}

4.3.1. Segment Duration and Distortions

First, in addition to the random segment durations ranging from 3-30 seconds, we also experimented with only segments of 30 seconds or full length audio (unsegmented). For either segmentation, speech activity boundaries were respected if possible, and all non-speech was also included.

Only two segmentation strategies were considered for UBM/T training: none or 120 second clips. UBM/T training was also considered with and without the included Switchboard data.

For the distortions, we tested two variations of the five individual distortions listed in Section 3.1. The specific parameters for each augmentation version are shown in Table 3, with the primary difference being that V2 (which was used in the official submission systems) uses milder settings than V1. In addition to these two mixtures, the effects of each individual distortion type were also tested (using the V2 parameters). In all cases, the clips were distorted according to the same process and proportions described in Section 3.1, except for the UBM/T list distortions, where the distorted files were appended to the training list rather than replacing unaugmented entries.

4.3.2. Results

The results are shown in Table 4 for various lists for both classifier training and UBM/T training. These results were generated for the BN joint i-vector system, though it should be noted these same analyses were run for our senone systems and an acoustic baseline, and the conclusions were similar in all cases.

Interestingly, the effects of segmentation and augmentation appear to be very different for classifier training and UBM/T

Table 4: C_{avg} for the BN joint i-vector system ([0] in Table 2) for several UBM/T training lists (columns) and several classifier training lists (rows), with the submission score italicized and the best combination in bold. Manipulations appear to be very valuable for classifier lists, but provide no gain and often degradation for the UBM/T lists.

Seg	Aug	Full	AugV1	Seg120	Full+Swb	AugV1+Swb	Seg120+Swb
None	None	19.31	21.91	21.36	27.59	26.22	26.99
30	None	18.50	19.38	19.19	20.72	20.12	20.31
30	V1	18.38	18.86	18.42	19.27	19.26	19.21
30	V2	18.80	19.14	18.74	19.95	19.57	19.78
Random	None	19.41	19.28	19.11	19.74	19.41	19.70
Random	V1	**18.35**	18.41	18.45	18.88	18.63	19.07
Random	V2	18.50	18.50	18.50	19.18	18.90	*19.24*

Table 5: Results for the individual augmentations applied to the random-length segments using the V2 parameters from Table 3. All augmentations improve performance, with the phone channel encoding providing the most gain.

Augmentation	C_{avg}
None	19.41
Resample only	18.97
Noise	18.78
Reverb	18.88
Compress	18.87
Phone	18.58
V2	18.50

training. In the case of the latter, the best list for UBM/T training includes only the LRE training data with no augmentation or segmentation and without the inclusion of the Switchboard data. This is particularly interesting, because this list also includes the fewest separate files for training, which runs counter to standard intuition for such training. This result may indicate the skewing of the language distribution from the Switchboard data was detrimental, or it may be that augmentation is less meaningful for unsupervised training.

For classifier training, on the other hand, it is clear that both duration segmentation and augmentation do provide value. Without any augmentation, the two segmentation types are slightly different, though the relative orientation changes depending on the UBM/T list used. However, once augmentations are included, the two segmentation strategies are frequently barely distinguishable. Also, the slightly more aggressive V1 augmentation performs similarly to V2.

These list comparisons also clearly show that the classifier augmentation selections for the official LRE submission are the best choices, and the gift of hindsight allows for a decrease in C_{avg} of roughly 0.9 absolute, from 19.24 to 18.35, mostly by improving the UBM/T training list. Table 2 shows the results for the same list combinations as compared to the submission lists, and similar magnitude gains are seen for all systems. It can also be seen in Table 2 that, after including all post-evaluation improvements, the score for the primary fusion submission is reduced to 17.32 (from an 18.54 submission score and 18.01 after classifier modifications).

4.3.3. Augmentation Types

Results were also considered when only including a single type of augmentation in order to better understand the contribution of each. This was tested for the augmentation options from V2 shown in Table 3, and results measured in C_{avg} are shown in Table 5 using the unsegmented and unaugmented UBM/T training list that was found to be the best performing option in the previous analysis.

A few clear conclusions can be drawn from the results of the experiments. First, inclusion of any of the augmentations improves performance. This finding is significant, because the across-the-board improvements provide a strong statement in support of augmentation for language recognition. While these are only a selected set of possible augmentation types and are only evaluated at a particular set of parameters, the augmentation improved performance in all cases.

Within these augmentation types, phone channel encoding appears to provide the largest gains in performance, which is reasonable considering the prevalence of phone data in the evaluation. Still, the combination of the augmentation types leads to further gains beyond the phone channel (or any individual augmentation) alone.

4.3.4. Language Dependence

Scores for each language cluster are also shown in Table 6, which provides some opportunity to understand the nature of the effects of augmentation. The first row shows results for no augmentation at all, the second row is for the submission lists, and the final row shows results using the augmentation strategies shown to be best in the post-evaluation analysis. The first conclusion that can be drawn from these results is that the augmentations help for most clusters, with the exception of French and Iberian (SPA). This is somewhat surprising, because the intuition for augmentation had been that it would help for languages with limited training data, which is the case for both the French and Iberian clusters. Yet, augmentation does still help for resource-rich clusters such as Arabic or Slavic. While the improvements to the English and Chinese clusters match with intuition, the remaining results make it clear that the effects of augmentation are not as simple as only increasing the number of training trials. These results may better be explained as increased robustness to channel mismatch, but a better understanding of the evaluation content would be required to prove this relationship.

Table 6: Effects of augmentation via resegmentation and distortion for each language cluster. Results are shown for the bottleneck joint i-vector system.

	ARA	CHN	ENG	FRE	SPA	SLA
Full; Full	22.90	17.50	10.81	42.57	17.69	4.40
Submission	20.91	12.99	8.82	49.80	18.95	4.00
Full; random AugV1	19.79	11.60	8.16	49.09	17.89	3.54

5. Conclusion

In this paper, we outlined the JHU HLTCOE submission systems to NIST LRE15, which were among the top performers in the evaluation. In the course of describing these submissions, we introduced a new joint i-vector framework that fuses the complimentary views of acoustic and phonotactic i-vectors. We also explored the contributions of various augmentations to this particular language recognition task and found them all to be beneficial to varying degrees. Finally, through post-evaluation analysis, we found improved classifier parameters, training lists, and augmentation settings which reduced the primary fusion submission system C_{avg} from 18.54 to 17.32.

6. References

[1] "The NIST year 2015 language recognition evaluation plan," `http://www.nist.gov/itl/iad/mig/upload/LRE15_EvalPlan_v23.pdf`, 2015.

[2] N. Dehak, P. Kenny, R. Dehak, P. Ouellet, and P. Dumouchel, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 19, pp. 788–798, May 2011.

[3] Pavel Matejka, Le Zhang, Tim Ng, HS Mallidi, Ondrej Glembek, Jeff Ma, and Bing Zhang, "Neural network bottleneck features for language identification," *Proc. of IEEE Odyssey*, pp. 299–304, 2014.

[4] Fred Richardson, Douglas Reynolds, and Najim Dehak, "Deep neural network approaches to speaker and language recognition," *Signal Processing Letters, IEEE*, vol. 22, no. 10, pp. 1671–1675, 2015.

[5] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[6] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep neural networks for extracting Baum-Welch statistics for speaker recognition," in *Odyssey: The Speaker and Language Recognition Workshop*, 2014.

[7] D. Garcia-Romero and A. McCree, "Insights into deep neural networks for speaker recognition," in *Interspeech*, 2015.

[8] Y. Lei, L. Ferrer, A. Lawson, M. McLaren, and N. Scheffer, "Application of convolutional neural networks to language identification in noisy conditions," in *Proc. Odyssey*, 2014, pp. 287–292.

[9] A. McCree and D. Garcia-Romero, "DNN Senone MAP Multinomial i-vectors for Phonotactic Language Recognition," in *Proc. Interspeech*, 2015, pp. 394–397.

[10] Navdeep Jaitly and Geoffrey E. Hinton, "Vocal Tract Length Perturbation (VTLP) imporves speech recognition," in *Proceedings of the International Conference on Machine Learning*, 2013.

[11] Anton Ragni, Kate M. Knill, Shakti P. Rath, and Mark J. F. Gales, "Data augmentation for low resource languages," in *Proceedings of Interspeech*, 2014.

[12] Xiaodong Cui, Vaibhava Goel, and Brian Kingsbury, "Data Augmentation for Deep Convolutional Neural Network Acoustic Modeling," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2015.

[13] Tom Ko, Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "Audio Augmentation for Speech Recognition," in *Proceedings of Interspeech*, 2015.

[14] M. Kockmann, L. Burget, O. Glembek, L. Ferrer, and J. Černocký, "Prosodic speaker verification using subspace multinomial models with intersession compensation," in *Proc. Interspeech*, 2010, pp. 1061–1064.

[15] D. Povey et al., "The subspace Gaussian mixture model–A structured model for speech recognition," *Computer Speech & Language*, vol. 25, no. 2, pp. 404–439, 2011.

[16] A. McCree, "Multiclass discriminative training of i-vector language recognition," in *Proc. Odyssey*, 2014, pp. 166–172.

[17] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proc. Interspeech*, 2011, pp. 249–252.

[18] X. Zhang, J. Trmal, D. Povey, and S. Khudanpur, "Improving deep neural network acoustic models using generalized maxout networks," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[19] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," in *International Conference on Learning Representations (ICLR)*, 2015.

[20] Y. Zhang, E. Chuangsuwanich, and J. Glass, "Extracting deep neural network bottleneck features using low-rank matrix factorization," in *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, 2014.

[21] D. Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matejka, "Language recognition in ivectors space," in *Proc. Interspeech*, 2011, pp. 861–864.

[22] A. McCree, F. Richardson, E. Singer, and D. Reynolds, "Beyond frame independence: Parametric modeling of time duration in speaker and language recognition," in *Proc. Interspeech*, 2008, pp. 767–770.