Spoken Language Recognition Based on Senone Posteriors

Luciana Ferrer [1,2], *Yun Lei* [1], *Mitchell McLaren* [1], *Nicolas Scheffer* [1]

[1] Speech Technology and Research Laboratory, SRI International, California, USA
[2] Departamento de Computación, FCEN, Universidad de Buenos Aires and CONICET, Argentina

{lferrer,yunlei,mitch,scheffer}@speech.sri.com

Abstract

This paper explores in depth a recently proposed approach to spoken language recognition based on the estimated posteriors for a set of senones representing the phonetic space of one or more languages. A neural network (NN) is trained to estimate the posterior probabilities for the senones at a frame level. A feature vector is then derived for every sample using these posteriors. The effect of the language used in training the NN and the number of senones are studied. Speech-activity detection (SAD) and dimensionality reduction approaches are also explored and Gaussian and NN backends are compared. Results are presented on heavily degraded speech data. The proposed system is shown to give over 40% relative gain compared to a state-of-the-art language recognition system at sample durations from 3 to 120 seconds.

1. Introduction

Spoken language recognition (SLR) approaches can be divided in two families: the acoustic ones, based on information extracted from short-term features like Mel-frequency cepstral coefficients (MFCCs), shifted delta cepstrum (SDC), and so on; and the phonotactic ones, based on phone information extracted using speech recognizers. The first family aims at directly describing the acoustic space of each language through the modeling of short-term spectral features. The second family leaves the job of acoustic modeling to an automatic speech recognizer which summarizes the information in the waveform into sequences of predicted phones. The distribution of phones and their sequences is then used to model the languages. A good review of both families of approaches along with a wealth of references can be found in [1].

One of the most successful approaches within the first family is the one based on i-vectors [2]. The i-vectors are modeled using a Gaussian backend (GB) or a NN. This approach will be used as baseline in our work.

Phonotactic approaches attempt to model the permissible combinations of phones in the languages of interest, and their frequencies. Standard phonotactic approaches involve collecting the probabilities for phone sequences as a representation of the signal using the output of one or several open-phone loop recognizers [3, 4, 5]. Language models or support vector machines are then used to generate the final scores. Another phonotactic approach uses the phoneme posteriogram counts from the phone recognizer to create bigram conditional probabilities which are then used to create features for SLR (e.g., [6]). A more recent approach [7] proposed the use of phone log-likelihood ratios (PLLR) at the frame level to extract i-vectors which are then modeled using a GB. Even though this approach uses phone recognition, it is not strictly phonotactic since it operates at the frame level and does not model phone sequences.

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. "A" (Approved for Public Release, Distribution Unlimited)

The described phonotactic approaches work with a relatively small set of units (usually around 50) aimed at representing the individual phones of the language being modeled. Information about the frequency of different phone sequences is collected through n-gram generation.

In this work, we explore in depth a recently proposed phonotactic system that uses senones as the basic phonetic unit [8]. Senones are defined as tied states within context-dependent phones. As such, they encode phone sequence information. They are the units from which word pronunciations are built in state-of-the-art automatic speech recognition (ASR) systems. The best performing ASR systems use deep NNs to predict the posteriors for a few thousand senones at each frame from input features including relatively long context information. Our proposed system uses these posteriors to create a feature vector for language recognition that is then modeled using standard backend techniques. The system fundamentally differs from previous phonotactic approaches in that it inherently models sequence information without the need for n-gram computation.

While the proposed system is quite simple, there are still many variables to optimize. In this article, we explore the parameter space of the proposed approach, including the number of senones, the data used for NN training, the speech-activity detection (SAD) approach, a dimensionality reduction technique and the backend approach. The resulting best configuration significantly outperforms the one used in our original paper [8]. We present results on data released by the Defense Advanced Research Projects Agency's (DARPA) Robust Automatic Transcription of Speech (RATS) program. The RATS data is heavily degraded by time-varying channel distortions.

2. System Description

The following sections first explain what senones are and how they are obtained. The NN model used to obtain the posteriors for the senones is then described. Finally, the proposed features based on these posteriors and the backend used to obtain language recognition scores are described.

2.1. Senone Definition

The senones are defined as states within context-dependent phones. Senones are the unit for which observation probabilities are computed during ASR. The pronunciations of all words are represented by a sequence of senones $\mathcal{Q}$. In general, the senone set $\mathcal{Q}$ is automatically defined by a decision tree [9]. At every node, a question is asked from a predefined set that includes questions about the left or right context, the central phone and the state number. An example question could be: "Is the phone to the left of this central phone a nasal?" The decision tree is grown in a greedy top-down manner by selecting at each node the question that gives 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 taken as the final set of senones. An example of a senone could be: the first state of all triphones where the central phone is /iy/, the right context is a nasal and the left context is /t/.

2.2. Senone Posterior Estimation

Traditionally, a Gaussian mixture model (GMM) was used to model the likelihood of the senones $p(x|q)$ for ASR. Recent studies have shown that deep NNs (DNNs) can be used to estimate the senone posteriors $p(q|x)$ which are converted into likelihoods using Bayes rule, a practice that improved performance with respect to traditional GMM systems [10, 11]. The input features given to the DNN are (typically) log Mel-filterbank coefficients. Each target frame is generally accompanied by context information that includes several filter bank feature vectors around the target frame. The output layer of the DNN contains one node for each senone defined by the decision tree.

For noisy conditions, convolutional NNs (CNNs) were proposed as replacements for DNNs to improve robustness against frequency distortion, first in the field of image processing by [12, 13], and then in the field of speech recognition [14, 15, 16]. Since our test data is extremely noisy and distorted, in this work we use CNNs for senone posterior estimation. A CNN is a NN in which the first layer is composed of one or more convolutional filters followed by max-pooling. In ASR, the filters are defined with the same length as the total number of frames, preventing convolution 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 of each filter is a single vector whose components are obtained by taking a weighted sum of several rows of the input matrix. After the convolutional filters are applied, the resulting vectors go through a process called max-pooling by which the maximum value is selected from N adjacent elements. The output vectors of the different filters after max pooling are concatenated into a long vector that is then input to a traditional DNN.

The CNN or DNN is trained to predict the senone posteriors at the frame level. The senone labels for each frame needed for training the network are obtained by force aligning the training data transcribed as a sequence of senones using a preexisting ASR model (usually a HMM-GMM system).

2.3. Features Definition

Figure 1 depicts the process of feature extraction. As described in the previous sections, given a speech sample i, a NN (DNN or CNN) can be used to generate the posteriors $\gamma_q(i,t)$ for every senone q and every frame t in the sample. The features are based on smooth counts provided by the NN, computed as

$$C_q(i) = \sum_{t\in T} \gamma_q(i,t),\ q \in Q \tag{1}$$

The set of frames T can be either all frames or only speech frames as defined by a separate SAD system. The set Q can include all senones or just those senones that correspond to speech states. The discarding of non-speech senones can be interpreted as a form of smooth SAD, since hard labeling of frames as speech or non-speech is not required.

The final features are obtained by normalizing the above counts and taking the logarithm:

$$Z_q(i) = \log\left(\frac{C_q(i)}{\sum_{s\in Q} C_s(i)}\right),\ q \in Q \tag{2}$$

If the set Q includes all senones, the denominator inside the log is simply the number of frames in set T since, in that case, $\sum_{q\in Q}\gamma_q(i,t) = 1$. On the other hand, if Q only contains speech frames, the denominator is a smooth measure of the number of speech frames within set T. In either case, the value inside the log is a probability distribution over q. This value can be interpreted as an estimation of the posterior probability of each senone in Q for the language present in sample i.

The dimension of the resulting vector is equal to the size of the set Q which can be much larger than the usual size of the i-vector modeled by standard backend approaches for SLR. For this reason, we explore the use of probabilistic principal component analysis (PPCA) to reduce the dimension of the feature vector. This strategy was proposed for our MLLR-based i-vector system in [17]. When PPCA is done, features are first normalized to have mean 0 and standard deviation 1 in each dimension.

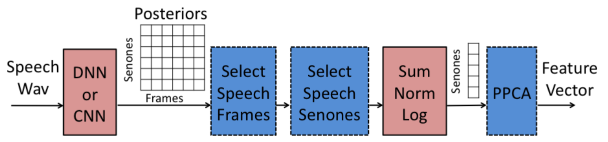

Figure 1: Feature extraction procedure for proposed system. Blue dashed boxes are optional. A NN is used to generate posteriors for each senone at each frame. Speech frames and speech senones are optionally selected. The sum over frames is then performed. The resulting smooth counts are normalized to sum to 1 and the log is computed. Optionally, a final step of PPCA is performed.

2.4. Backend

Two backends are compared in this work: a GB, the most standard backend for language recognition; and a NN. In the GB, a Gaussian distribution is estimated for each language with covariance shared across languages and language-dependent mean by maximizing the likelihood on the training data. The scores are computed as the likelihood of the samples given these Gaussian models.

The NN backend approach trains an NN to predict the posterior for each of the languages. Prior to NN modeling, the features are normalized to have mean 0 and standard deviation 1 on each dimension. The scores in this case are the weighted sums in the output layer, without applying the softmax function.

In this work, we focus on the language detection task where, for each test sample, the system has to answer the question: "Does this sample correspond to class X?" In our case, each test sample is tested against a small set of target languages and the out-of-set class corresponding to "non-target language." This class is trained using several non-target languages available in the training set. A final calibration step is done to transform the scores generated by the backend into likelihoods. This transformation is done using multiclass logistic regression as described in [18]. Finally, detection log-likelihood ratios (LLR) are computed from the likelihoods as described in [19]. Decisions are made by thresholding these LLRs at the theoretically optimal threshold.

3. Experiments

In this study, we evaluated the proposed approach on the RATS LID task consisting of five target languages (Farsi, Urdu, Pashto, Levantine Arabic and Dari) and a pre-defined set of ten out-of-set languages [20, 21]. 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 were considered in which test signals were constrained to have duration close to 3, 10, 30 and 120 seconds. The details of the task can be found in [22]. Tuning results are shown only for 3 and 30 seconds for lack of space.

The data used for training the UBM and i-vector models for the baseline system and the backends for all systems included five target languages and other out-of-set languages extracted from the RATS LID training set. Samples from this set were selected to constitute a relatively balanced distribution of languages, with a total of 23K segments with a mean speech duration of 80 seconds. This dataset was chunked into 8 and 30-second segments with 50% overlap. The chunked data and original data were used together for i-vector model, backend and PPCA training. The scores generated by the backends were further calibrated through multi-class logistic regression using 2-fold cross-validation on the test data.

The baseline system used in this study is the standard UBM/i-vector system followed by a NN backend. Similar to [23], a 140-dimensional 2D-DCT feature optimized for the RATS LID task is 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. The GMM-based SAD approach described in [24] was used for this system and for the experiments in Section 3.1 except power-normalized cepstral coefficients (PNCC) features [25] were used instead of MFCCs.

For the proposed approach, the RATS keyword-spotting training set including 260 hours of Levantine Arabic and 400 hours of Farsi data was used to train the HMM-GMMs and the CNNs. This data is used to train the CNNs because transcriptions are available for this data, while they are not available for the LID training data. The data includes clean and channel-degraded waveforms, which are treated independently; the relation between a clean waveform and the corresponding channel-degraded waveforms is not used anywhere in our system.

In Section 3.1 we compare CNNs of different sizes (small, medium and large), where the size is determined by the number of senones to be predicted. In Section 3.2 we compare CNNs trained with each language separately or with both languages and different number of senones. In all cases, the HMM-GMM ASR system was trained to maximize the likelihood of the training data. For the large CNNs the number of Gaussians in the HMM-GMM system is 300K, while for the smaller sizes is 200K. The features used in the HMM-GMM model trained with both languages were 39-dimensional MFCC features, including 13 static features (including C0) and first and second order derivatives. For historical reasons, the HMM-GMM trained with individual languages used a different set of features: 52-dimensional PLPs followed by HLDA to reduce dimension to 39. In all cases, these features were pre-processed with speaker-based cepstral mean and covariance normalization (MVN).

The CNNs were trained using cross entropy as error criteria on the alignments from these HMM-GMMs. The input features to the CNNs were given by 40 log Mel-filterbank coefficients with a context of 7 frames from each side of the center frame for which predictions were made. Two hundred convolutional filters of size 8 were used in the convolutional layer, and the pooling size was set to three without overlap. Five hidden layers of size 1200 and 2048 were used for the smaller CNNs and the large CNNs, respectively.

The performance was evaluated using the average detection cost over all target languages (Cavg) defined in the 2009 NIST Language Recognition Evaluation Plan [26], multiplied by 100.

3.1. SAD Approach

We first explore the approach for dealing with non-speech regions. Two approaches were compared: one in which we kept all frames, and one in which we discarded non-speech frames as determined by our SAD system. For each of those approaches, we also tried to either keep all senones or discard the three non-speech senones. The left plot in Figure 2 shows the comparison of the four approaches. A medium size CNN trained with Farsi and Levantine data was used for these experiments (see Section 3.2). No PPCA was performed, and the backend was an NN with 400 hidden nodes. The best approach for all durations is the one in which all frames are kept for computation of the smooth counts and the non-speech senones are discarded. The hard decisions made by the external SAD about the speech and non-speech frames could potentially discard too many frames. This hurts performance, especially for short durations. The rest of the experiments in this paper used this SAD approach, in which all frames are kept and the silence senones are discarded.

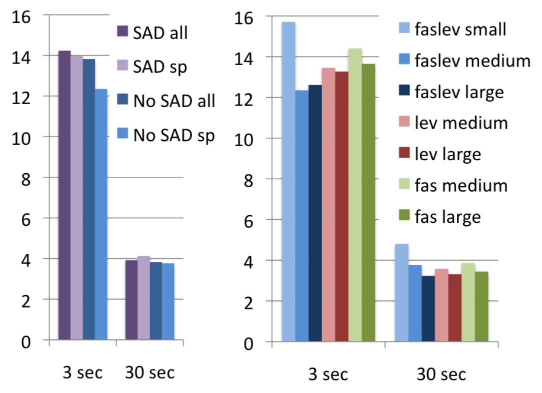

Figure 2: **Left:** Cavg$\times$100 for different SAD approaches which select frames using a SAD system (SAD) or not (No SAD), and keep all senones (all) or only speech senones (sp). The CNN trained with both languages (faslev) of medium size is used for these experiments. **Right:** Cavg$\times$100 for different CNNs trained with both available languages (faslev) or with only one of them (fas and lev) and varying number of senones (around 500, 3300 and 5500 for small, medium and large, respectively).

3.2. CNN Training Data and Number of Senones

Next, we explore the issue of the training data used for the CNN and the number of senones used. The right plot in 2 shows the performance of 7 CNNs: three trained on both Farsi and Levantine data with a different number of senones and two each trained only on Farsi or only Levantine data with only the larger number of senones. As in the previous section, no PPCA was performed, and the backend was a NN with 400 hidden nodes.

We can see that increasing the number of senones from the small to the medium size gives a significant gain in performance. A further increase in the size from the medium size to the large size gives a modest gain in most cases. We can also see that the combined use of both languages for training the CNN gives the best results at short durations.

3.3. Dimensionality Reduction and Backend Approach

Here, we explore the issue of dimensionality reduction using PPCA and the type, parameters, and training data for the backend. For the NN backend, we show results using a single hidden layer with varying number of nodes. The use of 0 or 2 hidden layers degrades performance (results not shown).

Figure 3 shows the performance for different PPCA dimensions and different number of hidden nodes. Results for the GB are also shown for comparison. In these experiments the PPCA matrix and the backend were trained with the original training data along with the 8 and 30 second chunks. The figure also shows the performance without PPCA using all the training data, including chunks, or only the original unchunked data for training. These experiments were done with the faslev CNN of medium size for run-time concerns.

The figure shows that the NN backend was always better than the GB for almost any PPCA size. We can also see that the use of PPCA degraded performance for any configuration of the backend. Furthermore, interestingly, an increase in the dimension for PPCA did not steadily improve performance toward the no-PPCA performance. We believe this is due to the fact that features generated by PPCA have the information concentrated in the first dimensions, with additional dimensions being less and less informative. The NN backend was apparently unable to completely ignore these noisy features. The number of nodes in the hidden layer has an inconsistent effect on performance, but a dimension of 400 seems to give a good trade-off for the no-PPCA case. Finally, results clearly show that use of the chunked data is essential for good performance at short durations. For 120 seconds, use of only the original data without chunks is slightly better than including the chunks. These results are not shown for lack of space.

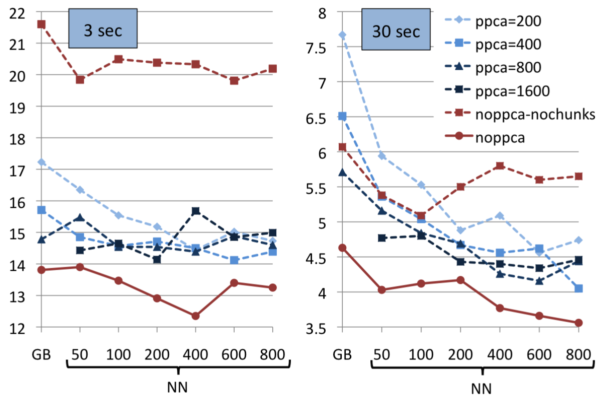

Figure 3: Cavg×100 for different PPCA dimensions and no PPCA for the GB and the NN with different number of hidden nodes. All experiments include the original data plus the 8 and 30 second chunks for training. For the setup without PPCA, results using only the original training data are also shown ("no chunks"). GB results for PPCA dimension of 1600 are missing, since the covariance matrix is extremely ill conditioned in that case.

3.4. Fusion Experiments

Finally, in Figure 4 we show the results for the baseline system and three proposed systems using different languages for training and the largest number of senones. No PPCA was performed, and the backend was an NN with 400 hidden nodes. The fusion is performed the same way as calibration, using linear logistic regression training the model on half of the test data and applying it to the other half and then rotating the sets.

We can see that the fusion, fas+lev, of the systems fas and lev where CNNs are trained with the individual languages gave significantly better performance than the faslev system where the CNN was trained with both languages. Addition of the faslev system to the fusion of fas and lev systems did not give further gains (result not shown). Furthermore, addition of the baseline to this fusion did not lead to consistent gains across durations. Overall, we see that the system that fused the fas and lev proposed systems performed between 43% and 62% better than the baseline system.

The system presented in our original paper [8] used both languages for CNN training, a medium number of senones, applied SAD for frame selection without discarding silence senones and used PPCA of dimension 400 with a NN with 200 hidden nodes. The fas+lev fusion in Figure 4 yielded 27% to 46% better performance over that original system.

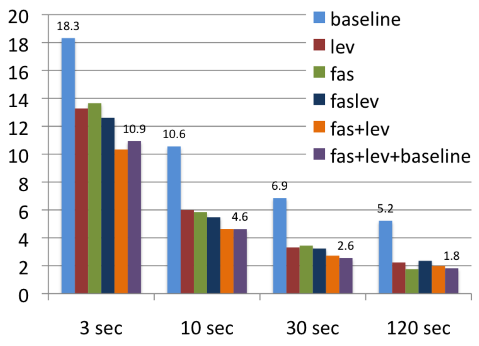

Figure 4: Cavg×100 for the baseline, different proposed systems and fusion of systems. Values for the baseline and the 3-way fusion are shown on top of the corresponding bars.

4. Conclusions

We explore a recently proposed front-end for language recognition that makes use of the posterior probabilities generated by an NN for a set of automatically defined senones. The posteriors over the frames are summed to compute smooth counts and normalized to generate a set of features which can then be modeled with standard backend approaches.

We study the approach in depth and conclude that an external SAD system is not necessary for optimal performance, that dimension reduction through PPCA is detrimental and that an NN backend with a single hidden layer of size around 400 is optimal in our test data. Most notably, we found that separate systems that use NNs trained with the individual languages available in training give better performance when fused than a system that uses a CNN trained with all available data. Overall, we show gains between 43% and 62% over different test durations on highly degraded data with respect to a state-of-the-art baseline system.

Further work includes testing the proposed approach on NIST language recognition evaluation data, comparison with a PPRLM system and early fusion approaches for the systems trained with individual languages.

5. References

[1] Haizhou Li, Bin Ma, and Kong Aik Lee, "Spoken language recognition: from fundamentals to practice," *Proceedings of the IEEE*, 2013.

[2] D. Gonzalez Martinez, O. Plchot, L. Burget, O. Glembek, and P. Matejka, "Language recognition in ivectors space," in *Proc. Interspeech*, Lyon, France, Aug. 2013.

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

[4] 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.

[5] A. Stolcke, M. Akbacak, L. Ferrer, S. Kajarekar, C. Richey, N. Scheffer, and E. Shriberg, "Improving language recognition with multilingual phone recognition and speaker adaptation transforms," in *Proc. Odyssey-10*, Brno, Czech Republic, June 2010.

[6] L. F. D'Haro, 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.

[7] M. Diez, A. Varona, M. Penagarikano, L. J. Rodriguez-Fuentes, and G. Bordel, "On the use of log-likelihood ratios as features in spoken language recognition," in *IEEE Workshop on Spoken Language Technology (SLT 2012), Miami, Florida, USA*, 2012.

[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-14*, Joensuu, Finland, June 2014.

[9] 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.

[10] 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.

[11] 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.

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

[13] 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.

[14] O. Abdel-Hamid, A. Mohamed, 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.

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

[16] 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.

[17] N. Scheffer, Y. Lei, and L. Ferrer, "Factor analysis back ends for MLLR transforms in speaker recognition," in *Proc. Interspeech*, Lyon, France, Aug. 2013.

[18] D. A. Van Leeuwen and N. Brummer, "Channel-dependent GMM and multi-class logistic regression models for language recognition," in *Proc. Odyssey-06*, Puerto Rico, USA, June 2006.

[19] N. Brummer and D. A. van Leeuwen, "On calibration of language recognition scores," in *Proc. Odyssey-06*, Puerto Rico, USA, June 2006.

[20] Aaron Lawson, Mitchell McLaren, Yun Lei, Vikramjit Mitra, Nicolas Scheffer, Luciana Ferrer, and Martin Graciarena, "Improving language identification robustness to highly channel-degraded speech through multiple system fusion," in *Proc. Interspeech*, Lyon, France, Aug. 2013.

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

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

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

[24] M. McLaren, N. Scheffer, M. Graciarena, L. Ferrer, and Y. Lei, "Improving speaker identification robustness to highly channel-degraded speech through multiple system fusion," in *Proc. ICASSP*, Vancouver, May 2013.

[25] C. Kim and R.M. Stern, "Power-normalized cepstral coefficients (PNCC) for robust speech recognition," in *Proc. ICASSP*, Kyoto, Mar. 2012.

[26] "NIST LRE09 evaluation plan," http://www.itl.nist.gov/iad/mig/tests/lre/2009/LRE09_EvalPlan_v6.pdf.