END-TO-END DNN BASED SPEAKER RECOGNITION INSPIRED BY I-VECTOR AND PLDA

Johan Rohdin, Anna Silnova, Mireia Diez, Oldřich Plchot, Pavel Matějka, Lukáš Burget

Brno University of Technology, Brno, Czechia

{rohdin, isilnova, mireia, iplchot, matejkap, burget}@fit.vutbr.cz

ABSTRACT

Recently, several end-to-end speaker verification systems based on deep neural networks (DNNs) have been proposed. These systems have been proven to be competitive for text-dependent tasks as well as for text-independent tasks with short utterances. However, for text-independent tasks with longer utterances, end-to-end systems are still outperformed by standard i-vector + PLDA systems. In this work, we develop an end-to-end speaker verification system that is initialized to mimic an i-vector + PLDA baseline. The system is then further trained in an end-to-end manner but regularized so that it does not deviate too far from the initial system. In this way we mitigate overfitting which normally limits the performance of end-to-end systems. The proposed system outperforms the i-vector + PLDA baseline on both long and short duration utterances.

Index Terms— Speaker verification, DNN, end-to-end

1. INTRODUCTION

In recent years, there have been many attempts to take advantage of neural networks (NNs) in speaker verification. Most of the attempts have replaced or improved one of the components of an i-vector + PLDA system (feature extraction, calculation of sufficient statistics, i-vector extraction or PLDA) with a neural network. For example by using NN bottleneck features instead of conventional MFCC features [1], NN acoustic models instead of Gaussian mixture models for extraction of sufficient statistics [2], NNs for either complementing PLDA [3, 4] or replacing it [5]. More ambitiously, NNs that take the frame level features of an utterance as input and directly produce an utterance level representation, usually referred to as an *embedding*, have recently been proposed [6, 7, 8, 9, 10, 11]. The embedding is obtained by means of a *pooling mechanism*, for example taking the mean, over the framewise outputs of one or more layers in the NN [6], or by the use of a recurrent NN [7]. One effective approach is to train the NN for classifying a set of training speakers, i.e., using multiclass training [6, 10, 11]. In order to do speaker verification, the embeddings are extracted and used in a standard backend, e.g., PLDA. Ideally the NNs should however be trained directly for the speaker verification task, i.e., binary classification of two utterances as a *target* or a *non-target* trial [7, 8, 9]. Such systems are known as *end-to-end* systems and have been proven competitive for text-dependent tasks [7, 8] as well as text-independent tasks with short test utterances and an abundance of training data [9]. However, on text-independent tasks with longer utterances, end-to-end systems are still being outperformed by standard i-vector + PLDA systems [9].

One reason that end-to-end training has not yet been effective for long utterances in text-independent speaker verification could be overfitting on the training data. A second reason could be that the previous works have trained the NN on short segments even when long segments are used in testing. This reduces the memory requirements in training and reduces the risk of overfitting but introduces a mismatch between the training and test conditions.

In this work, we develop an end-to-end speaker verification system that is initialized to mimic an i-vector + PLDA baseline. The system consists of a NN module for extraction of sufficient statistics (**f2s**), an NN module for extraction of i-vectors (**s2i**) and finally, a discriminative PLDA (DPLDA) model [12, 13] for producing scores. These three modules are first developed individually so that they mimic the corresponding part of the i-vector + PLDA baseline. After the modules have been trained individually they are combined and the system is further trained in an end-to-end manner on both long and short utterances. During the end-to-end training, we regularize the model parameters towards the initial parameters so that they do not deviate too far from them. In this way the system is prevented from becoming too different from the original i-vector + PLDA baseline which reduces the risk of overfitting. Additionally, by first developing the three modules individually, we can more easily find their optimal architectures as well as detect difficulties to be aware of in the end-to-end training.

We evaluate the system on three different data sets that are derived from previous NIST SREs. The three test sets contain speech from various languages and were designed to test the performance both on long (longer than two minutes) and short (shorter than 40s) utterances. The achieved results show that the proposed system outperforms both generatively and discriminatively trained i-vector + PLDA baselines.

2. DATASETS AND BASELINE SYSTEMS

2.1. Datasets

We followed the design of the PRISM [14] dataset in the sense of splitting the data into **training** and test sets. The PRISM set contains data from the following sources: NIST SRE 2004 - 2010 (also known as MIXER collections), Fisher English and Switchboard. During training of the end-to-end system initialization, we used the female portion of the NIST SRE'10 telephone condition (condition 5) to independently tune the performance of the blocks A and B in Figure 1.

We report results on three different datasets:

The work was supported by European Unions Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No. 748097, the Marie Sklodowska-Curie cofinanced by the South Moravian Region under grant agreement No. 665860, Google Faculty Research Award program, Czech Science Foundation under project No. GJ17-23870Y, Technology Agency of the Czech Republic project No. TJ01000208 "NOSICI", by a contract with NTT Corporation and by Czech Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project "IT4Innovations excellence in science - LQ1602"

- The female part of the **PRISM language** condition[1] that is based on original (long) telephone recordings from NIST SRE 2005 - 2010. It contains trials from various languages, including cross-language trials.
- The **short lang** condition (also containing only female trials) is derived from the PRISM language condition by taking multiple short cuts from each original recording. Durations of the speech in the cuts reflect the evaluation plan for NIST SRE'16 - more precisely we based our cuts on the actual detected speech in the SRE'16 labeled development data. We chose the cuts to follow the uniform distribution:
  - Enrollment between 25-50 seconds of speech
  - Test between 3-40 seconds of speech

  We split the resulting set into two equally large disjoint sets where speakers do not overlap. We used one part as our **dev** set for tuning the performance of the DPLDA and the end-to-end system. The other part was used for evaluation only. It should be noted that, for simplicity, we test only on single-enrollment trials unlike in our SRE'16 system description where we include multi-enrollment trials [15].
- Additionally, we report the results on the single-enrollment trials of the NIST SRE'16 evaluation set (both males and females).

2.2. Generative and Discriminative Baselines

As features we used 60-dimensional spectral features (20 MFCCs, including C_0, augmented with their Δ and $\Delta\Delta$ features). The features were short-term mean and variance normalized over a 3 second sliding window.

Both PLDA and DPLDA are based on i-vectors [16] extracted by means of UBM with 2048 diagonal covariance components. Both UBM and i-vector extractor with 600 dimensions are trained on the **training** set. For training our generative (PLDA) and discriminative (DPLDA [12]) baseline systems, we used only telephone data from the **training** set and we also included short cuts derived from portion of our training data that comes from non-English or non-native-English speakers. The duration of the speech in cuts follows the uniform distribution between 10-60 seconds. The cuts comprise of 22766 segments out of total 85858. Finally, we augmented the training data with labeled development data from NIST SRE'16.

PLDA: We used the standard PLDA recipe, when i-vectors are mean (mean is calculated using all training data) and length normalized. Then the Linear Discriminant Analysis (LDA) is applied prior PLDA training, decreasing dimensions of i-vectors from 600 to 250. We did not perform any additional domain adaptation or score normalization. We also filtered the training data in such a way that each speaker has at least six utterances which reduces it to the total of 62994 training utterances.

Discriminative PLDA: The DPLDA baseline model was trained on the full batch of i-vectors by means of LBFGS optimizing the binary cross-entropy on the training data. We used the **dev** set to tune a single constant that is used for L2 regularization imposed on all parameters except the constant (k in Eq. 1).

All i-vectors were mean (mean was calculated using all training data available) and length normalized. After the mean normalization, we performed LDA, decreasing the dimensionality of vectors to 250. As an initialization of DPLDA training, we used a corresponding PLDA model. During the DPLDA training, we set the prior probability of target trials to reflect the SRE'16 evaluation operating point (exactly in the middle between the two operating points of SRE'16 DCF [17]).

[1]For detailed description, please see section B, paragraph 4 of [14].

3. PROPOSED END-TO-END DNN ARCHITECTURE

The proposed system is depicted in Figure 1. In the following subsections, we first describe each of the individual modules and then the complete end-to-end system (please see [18] for details). The system was implemented using the Theano library [19].

3.1. Features to sufficient statistics

The first module of the end-to-end system converts a sequence of feature vectors into sufficient statistics. We will denote this module as **f2s**. This module consists of a network that predicts a vector of GMM responsibilities (posteriors) for each frame of the input utterance (Block A in Figure 1), followed by a layer for pooling the frames into sufficient statistics. The network that predicts responsibilities consists of four hidden layers with sigmoid activation functions and a softmax output layer. All hidden layers have 1500 neurons while the output layer has 2048 elements which corresponds to the number of components in our baseline GMM-UBM. We train this network with stochastic gradient descent (SGD) to optimize the categorical cross-entropy with the GMM-UBM posteriors as targets.

As input to the network, the acoustic features described in Section 2.2 are preprocessed as follows. For each frame, a window of 31 frames around the current frame (i.e. ± 15 frames) is considered. In this window, the temporal trajectory of each feature coefficient is weighted by a Hamming window and projected into first 6 DCT bases (including C_0) [20]. This results in a $6 \times 60 = 360$-dimensional input to the network for each frame.

Once the network for predicting responsibilities is trained, we add the layer that produces sufficient statistics. The input to this layer is a matrix of frame-by-frame responsibilities coming from the previous softmax layer and a matrix of original acoustic features without any preprocessing. This layer is not trained but designed in such a way that it exactly reproduces the standard calculation of sufficient statistics used in i-vector extraction. It should be noted that, in principle, expanding the features should not be necessary in order to predict the GMM-UBM posteriors since these are calculated from original features. However, by using the expanded features, we hope that we can gain further improvements in the end-to-end training.

3.2. Sufficient statistics to i-vectors

The second module of the end-to-end system is trained to mimic the i-vector extraction from the sufficient statistics (Block B in Figure 1). We will denote this module as **s2i**. The input sufficient statistics were first converted into MAP adapted supervectors [21] (using a relevance factor of $r = 16$). To overcome the computational problems that would arise when using the 122880 dimensional supervector as input to the NN, the supervectors were projected by PCA into a 4000 dimensional space. The NN consists of two 600 dimensional hidden layers, with hyperbolic tangent (tanh) activation functions. The last layer of the NN is designed to produce length normalized 250 dimensional i-vectors. As training objective, we use the average cosine distance between NN outputs and LDA reduced and length-normalized reference i-vectors. The NN is trained with SGD and L1 regularization.

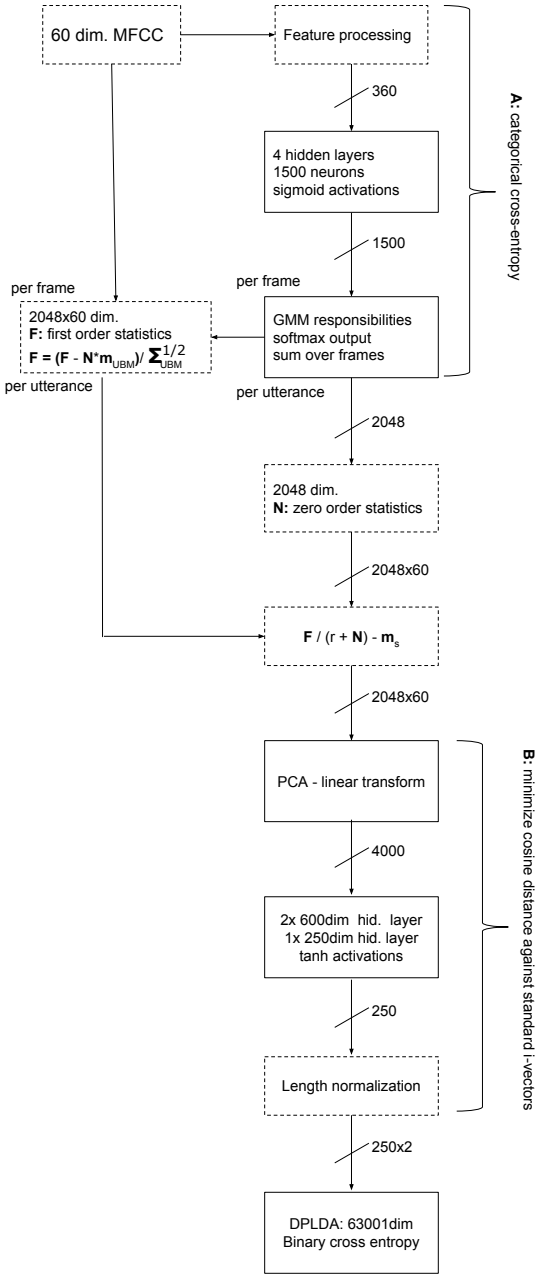

Fig. 1. Block diagram of the end-to-end system. Part **A** corresponds to the UBM that converts features to GMM responsibilities. By adding the next two blocks we obtain first order statistics (**f2s**). Part **B** (**s2i**) simulates the i-vector extraction followed by LDA and length normalization. Parameters in solid line blocks are meant to be trained, while outputs of the dashed blocks are directly computed.

3.3. i-vectors to scores (DPLDA)

The final component of the end-to-end system is a DPLDA model [12, 13]. Given two i-vectors $\boldsymbol{\phi}_i$ and $\boldsymbol{\phi}_j$, the LLR score for the PLDA model is given by

$$\begin{aligned} s_{ij} &= \boldsymbol{\phi}_i^T \boldsymbol{\Lambda} \boldsymbol{\phi}_j + \boldsymbol{\phi}_j^T \boldsymbol{\Lambda} \boldsymbol{\phi}_i + \boldsymbol{\phi}_i^T \boldsymbol{\Gamma} \boldsymbol{\phi}_i + \boldsymbol{\phi}_j^T \boldsymbol{\Gamma} \boldsymbol{\phi}_j \\ &+ (\boldsymbol{\phi}_i + \boldsymbol{\phi}_j)^T \boldsymbol{c} + k, \end{aligned} \quad (1)$$

where the parameters $\boldsymbol{\Lambda}$, $\boldsymbol{\Gamma}$, $\boldsymbol{c}$ and k can be calculated from the parameters of the PLDA model (see [12] for details). The idea of DPLDA is to train $\boldsymbol{\Lambda}$, $\boldsymbol{\Gamma}$, $\boldsymbol{c}$ and k directly for the speaker verification task, i.e., given two i-vectors, decide whether they are from the same speaker or not. This is achieved by forming trials (usually all possible) from the training data and optimizing, e.g., the binary cross-entropy or the SVM objective. In this work we use the binary cross-entropy objective.

DPLDA is trained iteratively and normally all training data is used in each iteration. Whenever the DPLDA model is trained individually in the experiments, we train it in this way. However, for an end-to-end system this would require too much memory and computational time. For this we therefore use minibatch training. It is not obvious how to optimally select the data for minibatches. In this paper, we used the following procedure:

1. For each speaker, randomly group his/her utterances into pairs.[2]
2. For each minibatch, randomly select (without replacement) N utterance pairs and use all trials that can be formed from these utterances. If the last pair is selected, repeat Step 1.

This approach gives batches with many speakers but with few utterances per speaker. Having more utterances per speaker in a batch would give us more target trials but these trials would have been statistically dependent which may affect the training negatively [22].

3.4. End-to-end system

After the individual components described in the previous subsections have been trained individually, they are combined to an end-to-end system. Unfortunately, combining the modules as they are leads to large memory requirements of the end-to-end system. This happens mainly for two reasons. First, contrary to the individual training of the modules, the PCA projection now needs to be part of the network in order for the **f2s** and **s2i** modules to be connected. The PCA matrix with 122880×4000 parameters uses approximately 2GB of memory. Second, the **f2s** now needs to process all frames from many different utterances in one batch to obtain the sufficient number of trials for the DPLDA module.

To mitigate the problem of the large PCA matrix we, before the complete end-to-end training, train only the **s2i** NN and the DPLDA model jointly. As for the individual training of **s2i**, we can use precalculated input that includes the PCA projection since this input is fixed as long as **f2s** is not updated. For this training we use minibatches of 5000 pairs (N). To mitigate the large memory requirements of the **f2s** module, we modify the training procedure to keep less intermediate results in memory. Specifically, in usual NN training, the input is first *forward propagated* through the network to get the output of each layer. These outputs are stored in memory and used during *backpropagation* to obtain the derivative of the loss with respect to each model parameter. For the part of **f2s** that calculates responsibilities (Block A in Figure 1), this results in $n_f(1500 + 1500 + 1500 + 1500 + 2048)$ variables to store in memory, where n_f is the total number of frames. This is much more than in subsequent modules (after pooling the frames into sufficient statistics, $\mathbf{F}$ and $\mathbf{N}$) where the layer outputs are per utterance. Thus, in order to reduce the memory usage, we calculate the sufficient statistics for one utterance at the time and discard all the layer outputs from Block A once the sufficient statistics for the utterance

[2]If a speaker has only one utterance, it will be used as a "pair". If a speaker has another uneven number of utterances, one of the "pairs" will be given three utterances.

Table 1. Overall results, $C_{\text{min}}^{\text{Prm}}$ and EER. Modules marked with a '*' are trained jointly. Other modules are trained sequentially.

	System Name	stats	i-vector	PLDA	SRE16		short lang		PRISM lang	
					$C_{\text{min}}^{\text{Prm}}$	EER	$C_{\text{min}}^{\text{Prm}}$	EER	$C_{\text{min}}^{\text{Prm}}$	EER
1	Baseline	UBM	i-extractor	Gen.	0.988	17.645	0.699	10.303	0.411	3.902
2	Baseline DPLDA	UBM	i-extractor	Discr.	0.975	16.902	0.616	9.462	0.360	3.461
3	f2s	NN	i-extractor	Gen.	0.980	16.809	0.687	9.866	0.394	3.713
4	s2i	UBM	NN	Gen.	0.988	16.686	0.788	11.141	0.430	4.584
5	f2s-s2i	NN	NN	Gen.	0.982	16.226	0.780	11.523	0.432	4.616
6	f2s-s2i-DPLDA	NN	NN	Discr	0.953	15.091	0.597	9.328	0.300	3.426
7	s2i-DPLDA_joint	NN	NN*	Discr.*	0.936	15.166	0.586	8.599	0.287	3.123
8	f2s-s2i-DPLDA_joint	NN*	NN*	Discr.*	0.936	15.170	0.587	8.661	0.287	3.125

have been calculated. When the sufficient statistics for all utterances have been obtained, we continue the forward propagation in the normal way, keeping all outputs in memory. During backpropagation, we recalculate the outputs of Block A when needed. This is achieved in a similar way as in `scan_checkpoints`[3]. This trick allows us to use minibatches of 75 pairs (N) instead of approximately 2.

Unlike the individual training of **f2s** and **s2i**, we use the ADAM optimizer [23] for training since it may be more robust to different learning rate requirements of the different modules compared to standard SGD. We halved the learning rate whenever we see no improvement in $C_{\text{min}}^{\text{Prm}}$ on the development set after an epoch (defined to be 250 batches). The training set was the same as for DPLDA.

4. RESULTS AND DISCUSSION

We report results in equal error rate (EER) as well as in the average minimum detection cost function for two operating points ($C_{\text{min}}^{\text{Prm}}$). The two operating points are the ones of interest in the NIST SRE'16 [17], namely the probability of target trials being equal to 0.01 and 0.005. Table 1 shows the results for the two baselines, the end-to-end system as well as systems where only some stages of the baseline have been replaced by a NN. Row 1 and Row 2 show the results for the PLDA and DPLDA baseline, respectively. The DPLDA performs better than generatively trained PLDA on all sets. This is consistent with our previous findings on NIST SRE'16 [15].

Row 3 shows the results when the UBM is replaced with the **f2s** NN. The i-vector extractor and PLDA model are trained as in the baseline but on the output of the **f2s** NN. It is noticeable that the **f2s** NN performs better than the UBM which it is supposed to mimic. The reason for this seems to be that the **f2s** NN is capable of learning a more robust model that generalizes better to the unseen data than the UBM, mainly because it uses a larger context. Our experiments in the development phase of the **f2s** NN showed that using the 60 dimensional features as input to a 2 layer **f2s** NN gave similar performance as the UBM ($C_{\text{min}}^{\text{Prm}}$ equal to 0.268 and 0.270 respectively on SRE'10, condition 5) whereas the large context features gave substantial improvement ($C_{\text{min}}^{\text{Prm}}$ equal to 0.254).

Row 4 shows the performance when i-vector extractor is replaced by the **s2i** NN. The input is the original statistics from the UBM and a PLDA model is trained on the output. We can see that, except for SRE'16, the performance degrades to some extent compared to the baseline (Row 1). Row 5 shows the results when we train a **s2i** module on the output from the **f2s** module instead of the statistics from the UBM. Again, we observe a small degradation compared to using a standard i-vector extractor (Row 3). Interestingly, when we further change from generative trained PLDA to DPLDA, the model performs better than both baselines. This suggests that the output from the **s2i** can well discriminate between speakers but may not well fulfill the PLDA model assumptions so that generative training does not work well.

After individual training of all blocks, we proceed with joint training of the **f2s** and **s2i** modules, using L2 regularization (tuned on the **dev** set) towards the parameters of the initial models. For this we use a batch size (N in Section 3.4) of 5000 pairs. As can be seen in the Row 7 of Table 1, the joint training of the two modules improves the performance on all data sets. Finally, the last row shows the performance when all modules are trained jointly. For this training, we can only use $N = 75$ as discussed in Section 3.3. As can be seen, the performance is almost unchanged from the previous row. There are three possible reasons for this. First, the minibatches might be too small for stable training. Second, with the **f2s** being well initialized and the subsequent modules already being trained to fit its output, the model may be stuck in a local minimum. Third, the **f2s** is in its current design quite constrained. It only estimates the responsibilities but cannot modify the features that are used to calculate the statistics. These issues will be studied in future work.

In summary, the final system achieved relative improvements with respect to the DPLDA baseline of 3.9%, 4.7% and 20.4% in $C_{\text{min}}^{\text{Prm}}$ on *SRE16*, *short lang* and *PRISM lang* respectively. In EER, the relative improvements were 10.2%, 8.5%, and 9.7%.

5. CONCLUSIONS AND FUTURE WORK

In this work, we have developed an end-to-end speaker verification system that outperforms an i-vector+PLDA baseline on three different datasets with utterances from many different languages and of both long and short durations. The system was constrained to behave similar to an i-vector + PLDA system. In this way we mitigated overfitting which normally limits the performance of end-to-end systems. This was a conservative approach and future work should explore if less constrained system can perform better, in particular as complement to i-vector+PLDA systems. We found that joint training of two modules of the three submodules of the system was effective but joint training all three modules was not effective. In future work we therefore want to develop more effective strategies for joint training of all three modules. The proposed system is designed for using single enrollment sessions, and extending it to deal with multiple enrollment sessions is also an important future work.

[3] http://www.deeplearning.net/software/theano/library/scan.html

6. REFERENCES

[1] A. Lozano-Diez, A. Silnova, P. Matějka, O. Glembek, O. Plchot, J. Pešán, L. Burget, and J. Gonzalez-Rodriguez, "Analysis and optimization of bottleneck features for speaker recognition," in *Proceedings of Odyssey 2016*. 2016, vol. 2016, pp. 352–357, International Speech Communication Association.

[2] Y. Lei, N. Scheffer, L. Ferrer, and M. McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 1695–1699.

[3] S. Novoselov, T. Pekhovsky, O. Kudashev, V. S. Mendelev, and A. Prudnikov, "Non-linear plda for i-vector speaker verification," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Sept 2015, pp. 214–218.

[4] G. Bhattacharya, J. Alam, P. Kenny, and V. Gupta, "Modelling speaker and channel variability using deep neural networks for robust speaker verification," in *2016 IEEE Spoken Language Technology Workshop, SLT 2016, San Diego, CA, USA, December 13-16*, 2016.

[5] O. Ghahabi and J. Hernando, "Deep belief networks for i-vector based speaker recognition," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 1700–1704.

[6] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2014, pp. 4052–4056.

[7] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 5115–5119.

[8] S. X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 171–178.

[9] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*, Dec 2016, pp. 165–170.

[10] G. Bhattacharya, J. Alam, and P. Kenny, "Deep speaker embeddings for short-duration speaker verification," in *Interspeech 2017*, 08 2017, pp. 1517–1521.

[11] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification," in *Interspeech 2017*, Aug 2017.

[12] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, and N. Brümmer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *Proc. of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, Prague, CZ, May 2011.

[13] S. Cumani, N. Brümmer, L. Burget, P. Laface, O. Plchot, and V. Vasilakis, "Pairwise discriminative speaker verification in the i–vector space," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 21, no. 6, pp. 1217–1227, june 2013.

[14] L. Ferrer, H. Bratt, L. Burget, H. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot, et al., "Promoting robustness for speaker modeling in the community: the prism evaluation set," *https://code.google.com/p/prism-set/*, 2012.

[15] O. Plchot, P. Matějka, A. Silnova, O. Novotný, M. Diez, J. Rohdin, O. Glembek, N. Brümmer, A. Swart, J. Jorrín-Prieto, P. García, L. Buera, P. Kenny, J. Alam, and G. Bhattacharya, "Analysis and Description of ABC Submission to NIST SRE 2016," in *Interspeech 2017*, Stockholm, Sweden, 2017.

[16] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. PP, no. 99, 2011.

[17] "The 2016 NIST speaker recognition evaluation plan (sre16)," https://www.nist.gov/file/325336.

[18] J. Rohdin, A. Silnova, M. Diez, O. Plchot, P. Matejka, and L. Burget, "End-to-end DNN Based Speaker Recognition Inspired by i-vector and PLDA," *ArXiv e-prints*, Oct. 2017.

[19] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[20] M. Karafiát, F. Grézl, K. Veselý, M. Hannemann, I. Szőke, and J. Černocký, "But 2014 babel system: Analysis of adaptation in nn based systems," in *Proceedings of Interspeech 2014*. 2014, pp. 3002–3006, International Speech Communication Association.

[21] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19–41, 2000.

[22] J. Rohdin, S. Biswas, and K. Shinoda, "Robust discriminative training against data insufficiency in plda-based speaker verification," *Computer Speech & Language*, vol. 35, pp. 32 – 57, 2016.

[23] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.