Deep Neural Network Embeddings for Text-Independent Speaker Verification

David Snyder, Daniel Garcia-Romero, Daniel Povey, Sanjeev Khudanpur

Center for Language and Speech Processing & Human Language Technology Center of Excellence,
The Johns Hopkins University, USA

{david.ryan.snyder, dpovey}@gmail.com, {dgromero, khudanpur}@jhu.edu

Abstract

This paper investigates replacing i-vectors for text-independent speaker verification with embeddings extracted from a feed-forward deep neural network. Long-term speaker characteristics are captured in the network by a temporal pooling layer that aggregates over the input speech. This enables the network to be trained to discriminate between speakers from variable-length speech segments. After training, utterances are mapped directly to fixed-dimensional speaker embeddings and pairs of embeddings are scored using a PLDA-based backend. We compare performance with a traditional i-vector baseline on NIST SRE 2010 and 2016. We find that the embeddings outperform i-vectors for short speech segments and are competitive on long duration test conditions. Moreover, the two representations are complementary, and their fusion improves on the baseline at all operating points. Similar systems have recently shown promising results when trained on very large proprietary datasets, but to the best of our knowledge, these are the best results reported for speaker-discriminative neural networks when trained and tested on publicly available corpora.

Index Terms: speaker recognition, speaker verification, deep neural networks

1. Introduction

Speaker verification (SV) is the task of authenticating the claimed identity of a speaker, based on some speech signal and enrolled speaker record. Typically, low-dimensional representations rich in speaker information are extracted for both enrollment and test speech, and compared to enable a same-or-different speaker decision. In modern systems, the representations are usually i-vectors. If the lexical content of the utterances is fixed to some phrase, the task is considered text-dependent, otherwise it is text-independent. This paper investigates replacing i-vectors with embeddings produced by a deep neural network (DNN) for text-independent SV. The relative strengths and weaknesses of this approach are assessed under a variety of conditions. In some practical applications, verification must be performed using only a limited amount of test speech, either to avoid latency in an online application or due to limited availability. To supplement the core 2010 NIST speaker recognition evaluation (SRE), we construct a modified version in which the enrollment utterances are full-length, but the test utterances have been truncated to the first few seconds of speech. Finally, we assess performance on the Cantonese and Tagalog NIST SRE 2016, which combines short-duration test conditions with language-mismatch.

1.1. Speaker verification with i-vectors

Most text-independent SV systems are based on i-vectors [1]. The standard system consists of a pipeline of generative models, trained on independent subtasks: a universal background model (UBM) that is used to collect sufficient statistics, a large projection matrix to extract i-vectors, and a probabilistic linear discriminant analysis (PLDA) backend to compute a similarity score between i-vectors [2, 3, 4, 5, 6, 7].

Traditionally, the UBM is a Gaussian mixture model (GMM) trained on acoustic features. Recent work has shown that incorporating an ASR DNN acoustic model can improve the UBM's ability to model phonetic content [8, 9, 10, 11, 12]. However, this comes at the cost of greatly increased computational complexity compared to traditional systems [11]. In addition, the advantages of incorporating ASR DNNs into the i-vector pipeline have been largely isolated to English language speech; [13] found no benefit in a multi-language setting. For these reasons, we restrict the scope of study to traditional i-vector systems using GMMs.

1.2. Speaker verification with DNNs

It may be possible to produce more powerful SV systems by training them to directly discriminate between speakers. Some studies have investigated discriminatively training components of the i-vector system [14, 15]. Given their success in other areas of speech technology, a natural alternative is to use DNNs trained on speaker-discriminative tasks. In early systems, neural networks are trained to classify training speakers [16, 17] or in Siamese architectures to separate same-speaker and different-speaker pairs [18, 19, 20]. After training, frame-level features are extracted from the networks and used as input to Gaussian speaker models. However, we are not aware of any work suggesting that those methods are competitive with modern i-vector systems for text-independent SV.

Progress has been primarily concentrated in text-dependent SV on large proprietary datasets. In [21], a feed-forward DNN is trained to classify speakers at the frame-level, on the phrase "OK Google." After training, the softmax output layer is discarded and speaker representations (called d-vectors) are created by averaging hidden layer activations. [22] built on this approach for the same application, by training an end-to-end system to discriminate between same-speaker and different-speaker pairs.

Recently, [23] showed that an end-to-end system that jointly learns embeddings along with a similarity metric could outperform a traditional i-vector baseline for text-independent SV. However, the approach required a large number of in-domain training speakers to be effective. Our system is based on [23], but modified in an effort to improve performance on smaller, publicly available datasets. We split the end-to-end approach into two parts: a DNN to produce embeddings and a separate backend to compare pairs of embeddings. Finally, instead of training the system to separate same-speaker and different-speaker pairs, the DNN learns to classify training speakers.

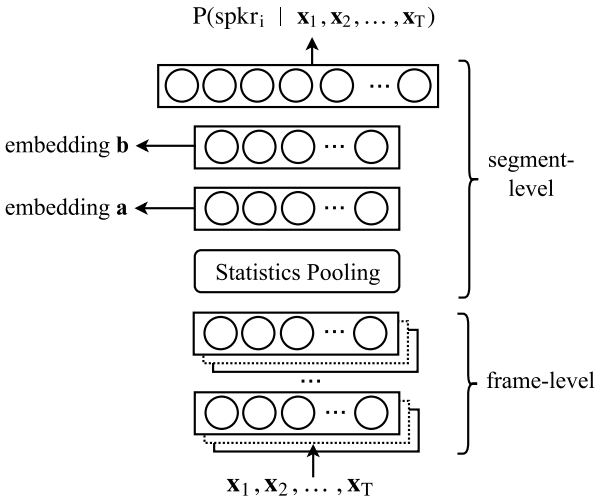

Figure 1: *Diagram of the DNN. Segment-level embeddings (e.g.,* **a** *or* **b***) can be extracted from any layer of the network after the statistics pooling layer.*

2. Baseline i-vector system

The baseline is a traditional i-vector system that is based on the GMM-UBM Kaldi recipe described in [11]. The front-end features consist of 20 MFCCs with a frame-length of 25ms that are mean-normalized over a sliding window of up to 3 seconds. Delta and acceleration are appended to create 60 dimension feature vectors. An energy-based VAD selects features corresponding to speech frames. The UBM is a 2048 component full-covariance GMM. The system uses a 600 dimension i-vector extractor. Prior to PLDA scoring, i-vectors are centered, dimensionality reduced to 150 using LDA, and length normalized. PLDA scores are normalized using adaptive s-norm [24].

3. DNN embedding system

3.1. Overview

The proposed system is a feed-forward DNN (depicted in Figure 1) that computes speaker embeddings from variable-length acoustic segments. The architecture is based on the end-to-end system described in [23]. However, an end-to-end approach requires a large amount of in-domain data to be effective. We replace the end-to-end loss with a multiclass cross entropy objective. In addition, a separately trained PLDA backend is used to compare pairs of embeddings. This enables the DNN and similarity metric to be trained on potentially different datasets. The network is implemented using the nnet3 neural network library in the Kaldi Speech Recognition Toolkit [25].

3.2. Features

The features are 20 dimensional MFCCs with a frame-length of 25ms, mean-normalized over a sliding window of up to 3 seconds. The same energy-based VAD from Section 2 filters out nonspeech frames. Instead of stacking frames at the input, short-term temporal context is handled by a time-delay DNN architecture.

3.3. Neural network architecture

The network, illustrated in Figure 1, consists of layers that operate on speech frames, a statistics pooling layer that aggregates over the frame-level representations, additional layers that operate at the segment-level, and finally a softmax output layer. The nonlinearities are rectified linear units (ReLUs).

The first 5 layers of the network work at the frame level, with a time-delay architecture [26]. Suppose t is the current time step. At the input, we splice together frames at $\{t-2, t-1, t, t+1, t+2\}$. The next two layers splice together the output of the previous layer at times $\{t-2, t, t+2\}$ and $\{t-3, t, t+3\}$, respectively. The next two layers also operate at the frame-level, but without any added temporal context. In total, the frame-level portion of the network has a temporal context of $t-8$ to $t+8$ frames. Layers vary in size, from 512 to 1536, depending on the splicing context used.

The statistics pooling layer receives the output of the final frame-level layer as input, aggregates over the input segment, and computes its mean and standard deviation. These segment-level statistics are concatenated together and passed to two additional hidden layers with dimension 512 and 300 (either of which may be used to compute embeddings) and finally the softmax output layer. Excluding the softmax output layer (because it is not needed after training) there is a total of 4.4 million parameters.

3.4. Training

The network is trained to classify training speakers using a multiclass cross entropy objective function (Equation 1). The primary difference between this and training in [16, 17, 21] is that our system is trained to predict speakers from variable-length segments, rather than frames. Suppose there are K speakers in N training segments. Then $P(spkr_k \mid \mathbf{x}_{1:T}^{(n)})$ is the probability of speaker k given T input frames $\mathbf{x}_1^{(n)}, \mathbf{x}_2^{(n)}, \ldots \mathbf{x}_T^{(n)}$. The quantity d_{nk} is 1 if the speaker label for segment n is k, otherwise it's 0.

$$E = -\sum_{n=1}^{N}\sum_{k=1}^{K} d_{nk} ln(P(spkr_k \mid \mathbf{x}_{1:T}^{(n)})) \quad (1)$$

The DNN is trained on the combined SWBD and SRE data described in Section 4.1. We refine the dataset by removing any recordings that are less than 10 seconds long, and any speakers with fewer than 4 recordings. This leaves a total of 4,733 speakers, which is the size of the softmax output layer.

To reduce sensitivity to utterance length, it is desirable to train the DNN on speech chunks that capture the range of durations we expect to encounter at test time (e.g., a few seconds to a few minutes). However, GPU memory limitations force a tradeoff between minibatch size and maximum training example length. As a comprise, we pick examples that range from 2 to 10 seconds (200 to 1000 frames) along with a minibatch size of 32 to 64. The example speech chunks are sampled densely from the recordings, resulting in about 3,400 examples per speaker. The network is trained for several epochs using natural gradient stochastic gradient descent [27].

3.5. Speaker embeddings

Ultimately, the goal of training the network is to produce embeddings that generalize well to speakers that have **not** been seen in the training data. We would like embeddings to capture speaker characteristics over the entire utterance, rather than at

the frame-level. Thus, any layer after the statistics pooling layer is a sensible place to extract the embedding from. We do not consider the presoftmax affine layer because of its large size and dependence on the number of speakers. In the network used in this work, we are left with two affine layers from which to extract embeddings. These are depicted in Figure 1 as embeddings $\mathbf{a}$ and $\mathbf{b}$. Embedding $\mathbf{a}$ is the output of an affine layer directly on top of the statistics. Embedding $\mathbf{b}$ is extracted from the next affine layer after a ReLU, and so it is a nonlinear function of the statistics. Since they are part of the same DNN, if embedding $\mathbf{b}$ is computed then we get embedding $\mathbf{a}$ for "free."

3.6. PLDA backend

We use the same backend for i-vectors and embeddings. Embeddings are centered and dimensionality is reduced using LDA. As in the i-vector system, we found that an LDA dimension of 25% of the original worked well. After dimensionality reduction, the embeddings are length normalized and pairs of embeddings are compared using PLDA. PLDA scores are normalized using adaptive s-norm [24]. As described in Section 3.5, the DNN architecture presents the option of using embeddings $\mathbf{a}$ or $\mathbf{b}$ or in combination. Instead of concatenating embeddings together, we compute separate PLDA backends for each embedding, and average the scores.

4. Experiments

4.1. Training data

The training data consists of telephone speech, the bulk of which is English. The *SWBD* portion consists of Switchboard 2 Phases 1, 2, and 3, and Switchboard Cellular. The *SRE* portion contains NIST SREs from 2004 through 2008. In total, there are about 65,000 recordings from 6,500 speakers. The i-vector UBM and extractor as well as the speaker discriminative DNN are trained on this data. Both systems use PLDA-based backends trained on just the SRE data. Finally, the 2016 NIST SRE was distributed with an unlabeled set of 2,472 utterances in Cantonese and Tagalog. For both systems, we use this to center the corresponding evaluation utterances and for score normalization.

4.2. Evaluation

We assess performance on NIST 2010 and 2016 speaker recognition evaluations [28, 29]. In the remaining sections, these will be abbreviated as *SRE10* and *SRE16* respectively. SRE10 consists of English telephone speech. Our evaluation is based on the extended core condition 5 and the 10s-10s condition. To supplement the core SRE10 condition, we produce additional conditions in which the enrollment utterances are full-length, but the test utterances have been truncated to the first $T \in \{5, 10, 20, 60\}$ seconds of speech, as determined by an energy-based VAD. The 10s-10s condition was part of the official SRE10 and consists of test **and** enrollment utterances that contain about 10 seconds of speech. SRE16 is comprised of Tagalog and Cantonese language telephone speech. The enrollment utterances contain about 60 seconds of speech while the test utterances range from 10 to 60 seconds of speech.

In addition to equal error-rate (EER), results are reported using the official performance metric for each SRE. For SRE10, this metric was the minimum of the normalized detection cost function (DCF) with $P_{\text{Target}} = 10^{-3}$ [28]. The primary SRE16 metric was a balanced (equalized) DCF averaged at two operating points [29]. The primary metrics are abbreviated to *DCF10* and *DCF16* respectively.

4.3. Results

In the following results, *ivector* refers to the traditional i-vector baseline described in Section 2. The labels *embedding* $\boldsymbol{a}$ and *embedding* $\boldsymbol{b}$ denote the systems consisting of embeddings extracted from either embedding layer of the same DNN (see Section 3.5) and used as features to their own PLDA backends. The label *embeddings* is the average of the PLDA backends for the individual embeddings. In the following results, we focus on comparing the i-vector baseline with these combined embeddings. Finally, *fusion* refers to the equally weighted sum fusion of the PLDA scores of *ivector* and *embeddings*.

4.3.1. NIST SRE10

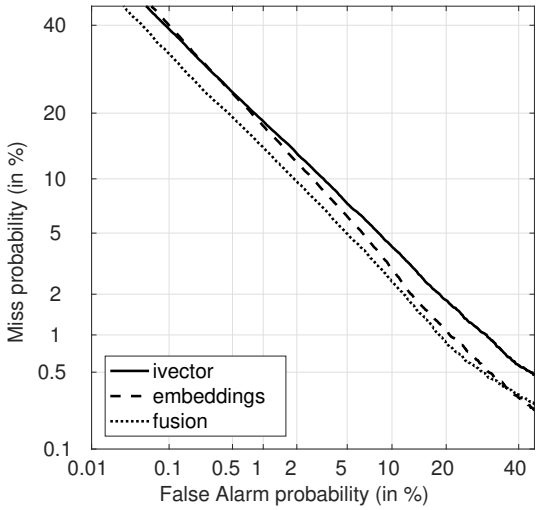

Figure 2: *DET curve for the pooled 5-60s portion of SRE10.*

Table 1: *EER(%) on NIST SRE10*

	10s-10s	5s	10s	20s	60s	full
ivector	11.0	9.1	6.0	3.9	2.3	1.9
embedding $\mathbf{a}$	11.0	9.5	5.7	3.9	3.0	2.6
embedding $\mathbf{b}$	9.2	8.8	6.6	5.5	4.4	3.9
embeddings	7.9	7.6	5.0	3.8	2.9	2.6
fusion	8.1	6.8	4.3	2.9	2.1	1.8

Table 2: *DCF10 on NIST SRE10*

	10s-10s	5s	10s	20s	60s	full
ivector	0.962	0.901	0.749	0.613	0.460	0.403
embedding $\mathbf{a}$	0.907	0.902	0.790	0.654	0.518	0.468
embedding $\mathbf{b}$	0.951	0.927	0.866	0.828	0.782	0.768
embeddings	0.854	0.875	0.738	0.667	0.567	0.539
fusion	0.859	0.788	0.645	0.556	0.432	0.383

In this section, we look at performance on the SRE10 con-

ditions described in Section 4.2. Tables 1 and 2 demonstrate the interplay between utterance-length and performance on SRE10. We see that i-vectors are still dominant for the longest recordings, and outperform embeddings at both the EER and DCF10 operating points. However, as the test utterance length decreases, the performance of the embeddings improves relative to the baseline. At 20 seconds of test speech, the combined embeddings are 3% better than i-vectors in EER but 8% worse at DCF10. With just 10 and 5 seconds of test speech, the embeddings are 17% and 16% better in EER and slightly better at DCF10. The relative advantage of embeddings appears to be largest when both enrollment and test utterances are short: in the column labeled *10s-10s* both the test **and** enroll utterances contain only about 10 seconds of speech, and we see that the combined embeddings are 28% better in EER and 11% better in DCF10. Figure 2 illustrates the detection error tradeoff (DET) curves for the systems when pooled across the truncated test conditions. Although embeddings are better at the EER operating point, they favor the low miss rate, and are slightly worse when compared at a very low false alarm rate.

Since the i-vector and DNN systems are so dissimilar, we expect good performance from their fusion. We observe an improvement over using i-vectors alone at all operating points and conditions. The largest improvement is in EER on the 10s condition, which is 28% better than the baseline. Even on the full-length condition, where i-vectors are strongest, there is a 5% improvement over using the i-vectors alone, in both DCF10 and EER.

4.3.2. NIST SRE16

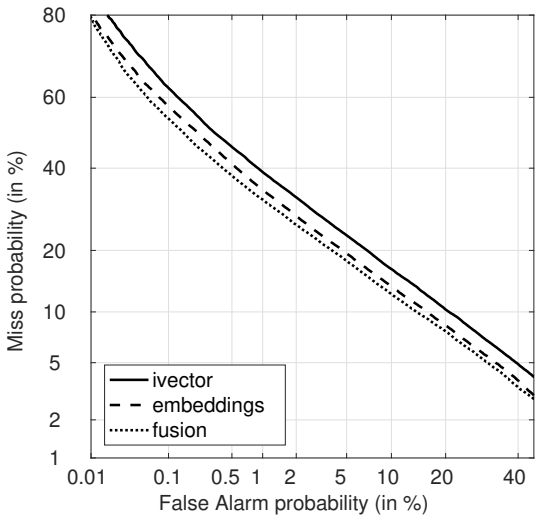

Figure 3: *DET curve for SRE16, pooled across Cantonese and Tagalog.*

In this section, we evaluate the same systems from Section 4.3.1 on SRE16. Using the same embedding and i-vector systems for both SRE10 and SRE16 avoids the complexity of developing variants of each system that are optimized for different evaluations. However, this does cause a mismatch between the predominately English training data (used to optimize both systems) and the Tagalog and Cantonese evaluation speech. As a result, the performance reported here may lag behind that of counterparts optimized specifically for SRE16.

Tables 3 and 4 report performance in EER and DCF16 respectively. Pooled across languages, we see that the combined embeddings outperforms the i-vector baseline by 13% in EER and 7% in DCF16. After combining with i-vectors, the improvement increases to 17% in EER and 13% in DCF16. The DET plot in Figure 3 shows that these improvements are consistent across operating points.

Table 3: *EER(%) on NIST SRE16*

	Cantonese	Tagalog	pool
ivector	8.3	17.6	13.6
embedding $\mathbf{a}$	7.7	17.6	13.1
embedding $\mathbf{b}$	7.8	17.4	13.1
embeddings	6.5	16.3	11.9
fusion	6.3	15.4	11.3

Table 4: *DCF16 on NIST SRE 2016*

	Cantonese	Tagalog	pool
ivector	0.549	0.842	0.711
embedding $\mathbf{a}$	0.532	0.835	0.689
embedding $\mathbf{b}$	0.630	0.851	0.741
embeddings	0.508	0.803	0.658
fusion	0.442	0.794	0.622

Although the embeddings also perform better on Tagalog, improvement is largest for the Cantonese portion. Compared to the i-vector baseline, the embeddings are 22% better in terms of EER and 7% in DCF16. The fused system is even better, and improves on the i-vector baseline by 24% in EER and 19% in DCF16.

5. Conclusions

In this paper, we investigated deep neural network embeddings for text-independent speaker verification. Overall, the embeddings appear to be competitive with a traditional i-vector baseline and are complementary when fused. We found that, although i-vectors are more effective on the full-length SRE10, embeddings are better on the short duration conditions. This underscores the findings of [23, 30] that DNNs may be capable of producing more powerful representations of speakers from short speech segments. SRE16 presented the challenge of language mismatch between the predominantly English training data and the Cantonese and Tagalog evaluation. We saw that embeddings outperformed i-vectors on both languages, suggesting that they may be more robust to this domain mismatch. Although the results are quite promising, we believe that PLDA may not be the optimal similarity metric for the embeddings. In future work, we will use the method described in this paper as pretraining for the fully end-to-end approach in [23], so that a more appropriate similarity metric is learned along with the embeddings.

6. Acknowledgments

This work was partially supported by NSF Grant No CRI-1513128, Nanyang Technological University (NTU) Science of Learning Grant, National Institutes of Standards and Technology Grant No 70NANB16H039. The authors thank Patrick Kenny and Paola Garcia for their helpful discussions.

7. References

[1] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, 2011.

[2] S. Prince and J. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007.*, Oct 2007, pp. 1–8.

[3] N. Brümmer and E. De Villiers, "The speaker partitioning problem." in *Odyssey*, 2010, p. 34.

[4] J. Villalba and N. Brümmer, "Towards fully bayesian speaker recognition: Integrating out the between-speaker covariance." in *INTERSPEECH*, 2011, pp. 505–508.

[5] P. Kenny, "Bayesian speaker verification with heavy-tailed priors." in *Odyssey*, 2010, p. 14.

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

[7] D. Garcia-Romero, X. Zhou, and C. Espy-Wilson, "Multicondition training of Gaussian plda models in i-vector space for noise and reverberation robust speaker recognition," in *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2012, pp. 4257–4260.

[8] P. Kenny, V. Gupta, T. Stafylakis, P. Ouellet, and J. Alam, "Deep neural networks for extracting Baum-Welch statistics for speaker recognition," in *Proc. Odyssey*, 2014.

[9] 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)*. IEEE, 2014, pp. 1695–1699.

[10] D. Garcia-Romero, X. Zhang, A. McCree, and D. Povey, "Improving speaker recognition performance in the domain adaptation challenge using deep neural networks," in *Spoken Language Technology Workshop (SLT), 2014 IEEE*. IEEE, 2014, pp. 378–383.

[11] D. Snyder, D. Garcia-Romero, and D. Povey, "Time delay deep neural network-based universal background models for speaker recognition," in *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*. IEEE, 2015, pp. 92–97.

[12] F. Richardson, D. Reynolds, and N. Dehak, "Deep neural network approaches to speaker and language recognition," *Signal Processing Letters, IEEE*, vol. 22, no. 10, pp. 1671–1675, 2015.

[13] O. Novotný, P. Matějka, O. Glembeck, O. Plchot, F. Grézl, L. Burget, and J. Černocký, "Analysis of the dnn-based sre systems in multi-language conditions," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016.

[14] O. Glembek, L. Burget, N. Brummer, O. Plchot, and P. Matejka, "Discriminatively trained i-vector extractor for speaker verification," in *INTERSPEECH*, 2011.

[15] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matějka, and N. Brümmer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *2011 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE, 2011, pp. 4832–4835.

[16] Y. Konig, L. Heck, M. Weintraub, and K. Sonmez, "Nonlinear discriminant feature extraction for robust text-independent speaker recognition," in *Proc. RLA2C, ESCA workshop on Speaker Recognition and its Commercial and Forensic Applications*, 1998.

[17] L. Heck, Y. Konig, K. Sonmez, and M. Weintraub, "Robustness to telephone handset distortion in speaker recognition by discriminative feature design," in *Speech Communication*, vol. 31, no. 2, 2000, pp. 181–192.

[18] K. Chen and A. Salman, "Learning speaker-specific characteristics with a deep neural architecture," in *IEEE Transactions on Neural Networks*, vol. 22, no. 11, 2011, pp. 1744–1756.

[19] ——, "Extracting speaker-specific information with a regularized siamese deep network," in *Advances in Neural Information Processing Systems (NIPS11)*, 2011.

[20] A. Salman, "Learning speaker-specific characteristics with deep neural architecture," Ph.D. dissertation, University of Manchester, 2012.

[21] E. Variani, X. Lei, E. McDermott, I. 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)*. IEEE, 2014, pp. 4052–4056.

[22] 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)*. IEEE, 2016, pp. 5115–5119.

[23] 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 *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016.

[24] S. Cumani, P. D. Batzu, D. Colibro, C. Vair, P. Laface, and V. Vasilakakis, "Comparison of speaker recognition approaches for real applications," in *INTERSPEECH*. ISCA, 2011.

[25] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlíček, Y. Qian, P. Schwarz *et al.*, "The Kaldi speech recognition toolkit," in *Proceedings of the Automatic Speech Recognition & Understanding (ASRU) Workshop*, 2011.

[26] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts." in *INTERSPEECH*, 2015, pp. 3214–3218.

[27] D. Povey, X. Zhang, and S. Khudanpur, "Parallel training of deep neural networks with natural gradient and parameter averaging," *CoRR*, vol. abs/1410.7455, 2015. [Online]. Available: http://arxiv.org/abs/1410.7455

[28] "The NIST year 2010 speaker recognition evaluation plan," http://www.itl.nist.gov/iad/mig/tests/sre/2010/, 2010.

[29] "NIST speaker recognition evaluation 2016," https://www.nist.gov/itl/iad/mig/speaker-recognition-evaluation-2016/, 2016.

[30] D. Garcia-Romero, D. Snyder, G. Sell, D. Povey, and A. McCree, "Speaker diarization using deep neural network embeddings," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 4930–4934.