Available online at www.sciencedirect.com

ScienceDirect

Computer Speech & Language 58 (2019) 403–421

www.elsevier.com/locate/csl

Analysis of DNN Speech Signal Enhancement for Robust Speaker Recognition[☆]

Ondřej Novotný*, Oldřich Plchot, Ondřej Glembek, Jan “Honza” Černocký, Lukáš Burget

Brno University of Technology, Speech@FIT and IT4I Center of Excellence, Božetěchova 2, Brno 612 66, Czech Republic

Received 16 November 2018; received in revised form 28 March 2019; accepted 2 June 2019
Available online 9 June 2019

Abstract

In this work, we present an analysis of a DNN-based autoencoder for speech enhancement, dereverberation and denoising. The target application is a robust speaker verification (SV) system. We start our approach by carefully designing a data augmentation process to cover a wide range of acoustic conditions and to obtain rich training data for various components of our SV system. We augment several well-known databases used in SV with artificially noised and reverberated data and we use them to train a denoising autoencoder (mapping noisy and reverberated speech to its clean version) as well as an x-vector extractor which is currently considered as state-of-the-art in SV. Later, we use the autoencoder as a preprocessing step for a text-independent SV system. We compare results achieved with autoencoder enhancement, multi-condition PLDA training and their simultaneous use. We present a detailed analysis with various conditions of NIST SRE 2010, 2016, PRISM and with re-transmitted data. We conclude that the proposed preprocessing can significantly improve both i-vector and x-vector baselines and that this technique can be used to build a robust SV system for various target domains.

Keywords: Speaker verification; Signal enhancement; Autoencoder; Neural network; Robustness; Embedding

1. Introduction

In recent years, there have been many attempts to take advantage of neural networks (NNs) in speaker verification (SV). They slowly found their way into the state-of-the-art systems that are based on modeling the fixed-length utterance representations, such as i-vectors (Dehak et al., 2011), by Probabilistic Linear Discriminant Analysis (PLDA) (Prince, 2007).

Most of the efforts to integrate the NNs into the SV pipeline involved replacing or improving one or more of the components of an i-vector + PLDA system (feature extraction, calculation of sufficient statistics, i-vector extraction or PLDA classifier) with a neural network. On the front-end level, let us mention for example using NN bottleneck features (BNF) instead of conventional Mel Frequency Cepstral Coefficients (MFCC, Lozano-Diez et al., 2016) or

[☆] This paper has been recommended for acceptance by Roger K. Moore.

* Corresponding author.
E-mail address: inovoton@fit.vutbr.cz (O. Novotný).

https://doi.org/10.1016/j.csl.2019.06.004
0885-2308/ 2019 Elsevier Ltd. All rights reserved.

simply concatenating BNF and MFCCs (Matějka et al., 2016) which greatly improves the performance and increases system robustness. Higher in the modeling pipeline, NN acoustic models can be used instead of Gaussian Mixture Models (GMM) for extraction of sufficient statistics (Lei et al., 2014) or for either complementing PLDA (Novoselov et al., 2015; Bhattacharya et al., 2016) or replacing it (Ghahabi and Hernando, 2014).

These lines of work have logically resulted in attempts to train a larger DNN directly for the SV task, i.e., binary classification of two utterances as a *target* or a *non-target* trial (Heigold et al., 2016; Zhang et al., 2016; Snyder et al., 2016; Rohdin et al., 2018). Such architectures are known as *end-to-end* systems and have been proven competitive for text-dependent tasks (Heigold et al., 2016; Zhang et al., 2016) as well as text-independent tasks with short test utterances and an abundance of training data (Snyder et al., 2016). In text-independent tasks with longer utterances and moderate amount of training data, the i-vector inspired end-to-end system (Rohdin et al., 2018) already outperforms generative baselines, but at the cost of high complexity in memory and computational costs during training.

While the fully end-to-end SV systems have been struggling with large requirements on the amount of training data (often not available to the researchers) and high computational costs, focus in SV has shifted back to generative modeling, but now with utterance representations obtained from a single NN. Such NN takes the frame level features of an utterance as an input and directly produces an utterance level representation, usually referred to as an *embedding* (Variani et al., 2014; Heigold et al., 2016; Zhang et al., 2016; Bhattacharya et al., 2017; Snyder et al., 2017). The embedding is obtained by the means of a *pooling mechanism* (for example taking the mean) over the frame-wise outputs of one or more layers in the NN (Variani et al., 2014), or by the use of a recurrent NN (Heigold et al., 2016). One effective approach is to train the NN for classifying a set of training speakers, i.e., using multiclass training (Variani et al., 2014; Bhattacharya et al., 2017; Snyder et al., 2017). In order to do SV, the embeddings are extracted and used in a standard backend, e.g., PLDA. Such systems have recently been proven superior to i-vectors for both short and long utterance durations in text-independent SV (Snyder et al., 2017; 2018).

Hand in hand with development of new modeling techniques that increase the performance of SV on particular benchmarks comes a requirement to continuously verify performance and improve robustness of the SV system under various scenarios and acoustic conditions. One of the most important properties of a robust system is the ability to cope with the distortions caused by noise and reverberation and by the transmission channel itself. In SV, one way is to tackle this problem in the late modeling stage and use multi-condition training (Martínez et al., 2014; Lei et al., 2012) of PLDA, where we introduce noise and reverberation variability into the within-class variability of speakers. This approach can be further combined with domain adaptation (Glembek et al., 2014) which requires having certain amount of usually unlabelled target data. In the very last stage of the system, SV outputs can be adjusted via various kinds of adaptive score normalization (Sturim and Reynolds, 2005; Matějka et al., 2017; Swart and Brümmer, 2017).

Another way to increase the robustness is to focus on the quality of the input acoustic signal and enhance it before it enters the SV system. Several techniques were introduced in the field of microphone arrays, such as active noise canceling, beamforming and filtering (Kumatani et al., 2012). For single microphone systems, front-ends utilize signal preprocessing methods, for example Wiener filtering, adaptive voice activity detection (VAD), gain control, etc. ETSI (2007). Various designs of robust features (Plchot et al., 2013) can also be used in combination with normalization techniques such as cepstral mean and variance normalization or short-time gaussianization (Pelecanos and Sridharan, 2006).

At the same time when DNNs were finding their way into basic components of the SV systems, the interest in NN has also increased in the field of signal pre-processing and speech enhancement. An example of classical approach to remove a room impulse response is proposed in Dufera and Shimamura (2009), where the filter is estimated by an NN. NNs have also been used for speech separation in Yanhui et al. (2014). A NN-based autoencoder for speech enhancement was proposed in Xu et al. (2014a) with optimization in Xu et al. (2014b) and finally, reverberant speech recognition with signal enhancement by a deep autoencoder was tested in the Chime Challenge and presented in Mimura et al. (2014).

In this work, we focus on improving the robustness of SV via a DNN autoencoder as an audio pre-processing front-end. The autoencoder is trained to learn a mapping from noisy and reverberated speech to clean speech. The frame-by-frame aligned examples for DNN training are artificially created by adding noise and reverberation to the Fisher speech corpus. Resulting SV systems are tested both on real and simulated data. The real data cover both telephone conversations (NIST SRE2010 and SRE2016) and speech recorded over various microphones (NIST SRE2010, PRISM, Speakers In The Wild - SITW). Simulated data are created to produce challenging conditions by either adding noise and reverberation into the clean microphone data or by re-transmission of the clean telephone and microphone data to obtain naturally reverberated data.

After we explore the benefits of DNN-based audio pre-processing with standard generative SV systems based on i-vectors and PLDA, we attempt to improve an already better baseline system where a DNN replaces the crucial i-vector extraction step. We use the architecture proposed by David Snyder (Snyder, 2017; Snyder et al., 2017), which already presents the *x-vector* (the embedding) as a robust feature for PLDA modeling, and provides state-of-the-art results across various acoustic conditions (Novotný et al., 2018b). We experiment with using the signal enhancement autoencoder as a pre-processing step while training the x-vector extractor, or just during the test stage. To further compare with the best i-vector system, we also experiment with using Stack Bottle-neck (SBN) features concatenated with MFCCs to train our x-vector extractor.

Finally, we offer experimental evidence and thorough analysis to demonstrate that the DNN-based signal enhancement increases the performance of text-independent SV system for both i-vector and x-vector based systems. We further combine the proposed method with multi-condition training that can significantly improve the SV performance and we show that we can profit from the combination of both techniques.

2. Speaker recognition systems

In this work, we compare four Speaker Verification systems (SV), combining two essential feature extraction techniques—MFCC, and Stack Bottle-neck features (SBNs) concatenated with MFCCs—and two front-end modelling techniques—the i-vectors and the x-vectors, defined in Matějka et al. (2014), Kenny (2010), Dehak et al. (2011) and Snyder et al. (2017). Please note, that each of the modeling techniques uses slightly different MFCC extraction.

SBN features were shown to be robust against various acoustic environments and using them in concatenation with MFCCs in simple i-vector framework not only brought the state-of-the-art performance, but also eliminated the need of performing more complicated alignment of GMM-UBM components via the deep neural network (Novotný et al., 2016). MFCC coefficients are therefore still important features that are not only used in combination with SBNs; the present-day state-of-the-art Kaldi x-vector systems utilize MFCCs alone. Another reason for presenting systems based on pure MFCCs are the results obtained in the recent NIST evaluations (NIST SRE 2016) where MFCCs alone were outperforming systems built atop concatenation of MFCCs and SBNs (Plchot et al., 2017).

All systems use the same voice activity detection (VAD) based on the BUT Czech phoneme recognizer[1], as described in Matějka et al. (2006), dropping all frames that are labeled as silence or noise. The recognizer was trained on Czech CTS data, but we have added noise with varying SNR to 30% of the database. This VAD was used both in the hyper-parameter training, as well as in the test phase.

Speaker verification scores were produced by comparing two i-vectors (or x-vectors) by the means of Probabilistic Latent Discriminant Analysis (PLDA) (Kenny, 2010).

2.1. MFCC i-vector system

In this system, we used cepstral features, extracted from 25 ms Hamming-windowed frames. We used 24 Mel-filters and the limited the band to 120–3800Hz range. 19 MFCCs together with zero-*th* coefficient were calculated every 10 ms. This 20-dimensional feature vector was subjected to short time mean- and variance-normalization using a 3 s sliding window. Delta and double delta coefficients were then calculated using a five-frame window, resulting in a 60-dimensional feature vector.

The acoustic modelling in this system is based on i-vectors. We use 2048-component diagonal-covariance Universal Background Model (GMM-UBM), and we set the dimensionality of i-vectors to 600. We then apply LDA to reduce the dimensionality to 200. Such i-vectors are then centered around a global mean followed by length normalization (Dehak et al., 2011; Garcia-Romero and Espy-Wilson, 2011).

2.2. SBN-MFCC i-vector system

Bottleneck Neural-Network (BN-NN) refers to such a topology of a NN, where one of the hidden layers has significantly lower dimensionality than the surrounding ones. A bottleneck feature vector is generally understood as a

[1] https://speech.fit.vutbr.cz/software/phoneme-recognizer-based-long-temporal-context

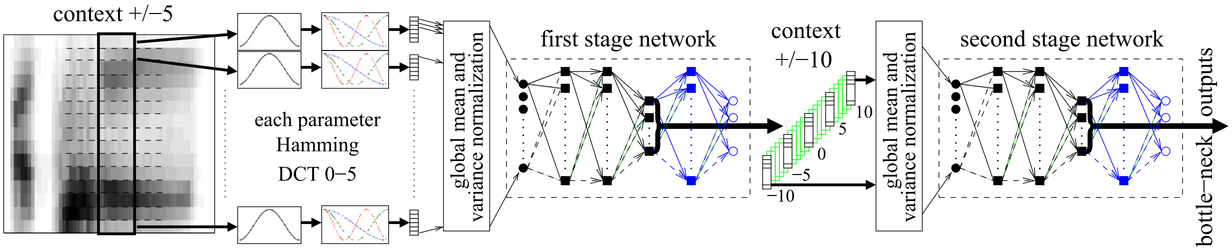


Fig. 1. Block diagram of Stacked Bottle-Neck (SBN) feature extraction. The blue parts of neural networks are used only during the training. The green frames in context gathering between the two stages are skipped. Only frames with shift -10, -5, 0, 5, 10 form the input to the second stage NN.

by-product of forwarding a primary input feature vector through the BN-NN and reading off the vector of values at the bottleneck layer. We have used a cascade of two such NNs for our experiments. The output of the first network is *stacked* in time, defining context-dependent input features for the second NN, hence the term Stacked Bottleneck features (SBN, Fig. 1).

The NN input features are 24 log Mel-scale filter bank outputs augmented with fundamental frequency features from 4 different f_0 estimators (Kaldi, Snack[2], and other two according to Laskowski and Edlund (2010) and Talkin (1995)). Together, we have 13 f_0 related features, see Karafiát et al. (2014) for details. Conversation-side based mean subtraction is applied on the whole feature vector, then 11 frames of log filter bank outputs and fundamental frequency features are stacked. Hamming window and DCT projection (0^{th} to 5^{th} DCT base) are applied on the time trajectory of each parameter resulting in $(24 + 13) \times 6 = 222$ coefficients on the first stage NN input.

The configuration of the first NN is $222 \times D_H \times D_H \times D_{BN} \times D_H \times K$, where $K = 9824$ is the number of target triphones. The dimensionality of the bottleneck layer, D_{BN} was set to 30. The dimensionality of other hidden layers D_H was set to 1500. The bottleneck outputs from the first NN are sampled at times $t-10$, $t-5$, t, $t+5$ and $t+10$, where t is the index of the current frame. The resulting 150-dimensional features are inputs to the second stage NN with the same topology as the first stage. The network was trained on the Fisher English corpus, and data were augmented with two noisy copies.

Finally, the 30-dimensional bottleneck outputs from the second NN (referred to as SBN) were concatenated with MFCC features (as used in the previous system) and used as an input to the conventional GMM-UBM i-vector system, with 2048 components in the UBM and 600-dimensional i-vectors.

2.3. The x-vector systems

These SV systems are based on a deep neural network (DNN) architecture for the extraction of embeddings as described in Snyder et al. (2017) and Snyder et al. (2018). Specifically, we use the original Kaldi recipe (Snyder, 2017) and 512 dimensional embeddings extracted from the first layer after the pooling layer (embedding-a, also referred to as the x-vector), which is consistent with Snyder et al. (2018).

Input features to the DNN were MFCCs, extracted from 25 ms Hamming-windowed frames. We used 23 Mel-filters and we limited the band to 20−3700 Hz range. 23 MFCCs were calculated every 10 ms. This 23-dimensional feature vector was subjected to short time mean- and variance-normalization using a 3 s sliding window. Note the differences to the MFCC features for i-vector system described above (mainly the number of Mel-filter banks, bandwidth, no delta/double delta coefficients).

The embedding DNN can be divided into three parts. The first part operates on the frame level and begins with 5 layers of time-delay architecture, described in Peddinti et al. (2015). The first four layers contain each 512 neurons, the last layer before statistics pooling has 1500 neurons. The following pooling layer gathers mean and standard deviation statistics from all frame-level inputs. The single vector of concatenated means and standard deviations is propagated through the rest of the network, where embeddings are extracted. This part consists of two hidden layers

[2] http://kaldi.sourceforge.net, http://www.speech.kth.se/snack/

each with 512 neurons and the final output layer has a dimensionality corresponding to the number of speakers. The DNN uses Rectified Linear Units (ReLUs) as nonlinearities in hidden layers, soft-max in the output layer and is trained by optimizing multi-class cross entropy.

In addition, we also trained an x-vector extractor on MFCC features concatenated with SBN from Section 2.2. Apart from changing the input features, we kept the architecture of the embedding DNN the same as for the MFCC system.

3. Signal enhancement autoencoder

For training the denoising autoencoder, we needed a fairly large amount of clean speech from which we formed a parallel dataset of clean and augmented (noisy, reverberated or both) utterances. We chose Fisher English database Parts 1 and 2 as they span a large number of speakers (11971) and the audio is relatively clean and without reverberation. These databases combined contain over 20,000 telephone conversational sides or approximately 1800 hours of audio.

Our autoencoder, introduced in Plchot et al. (2016) and in Novotný et al. (2018a), consists of three hidden layers with 1500 neurons in each layer. The input of the autoencoder is a 129-dimensional log-magnitude spectrum with a context of +/- 15 frames (in total 31 × 129 = 3999-dimensional input). The output is a 129-dimensional enhanced central frame log-magnitude spectrum, see the topology in Fig. 2.

It was necessary to perform feature normalization during the training and then repeat a similar process during actual denoising. We used the mean and variance normalization with mean and variance estimated per input utterance. At the output layer, de-normalization with parameters estimated on a clean variant of the file was used during training while during denoising, the mean and variance were global and estimated on the cross-validation set. Using log on top of the magnitude spectrum decreases the dynamic range of the features and leads to a faster convergence.

As an objective function for training the autoencoder, we used the Mean Square Error (MSE) between the autoencoder outputs from training utterances and spectra of their clean variants. We were using both clean and augmented (noisy) recordings during the training as we wanted the autoencoder to be both robust and producing good results on relatively clean data.

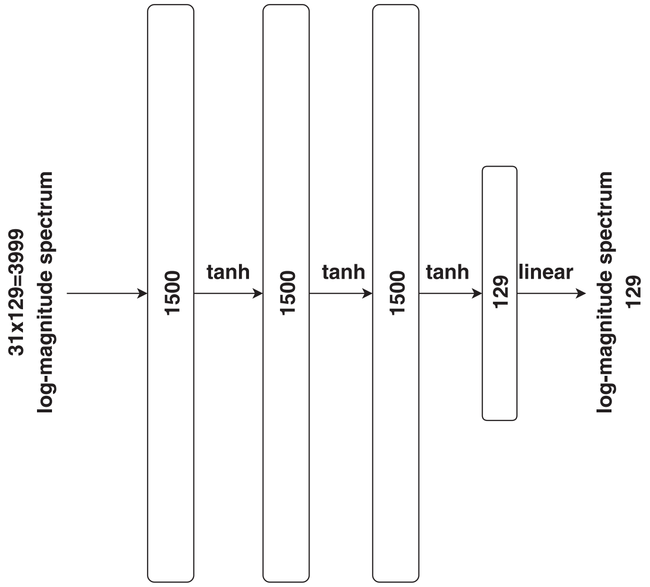


Fig. 2. Topology of the autoencoder: three hidden layers each with 1500 neurons and hyperbolic tangent activation functions, output layer with 129 neurons and linear activation functions. The input of the network are 31 concatenated frames of the 129-dimensional log-magnitude spectrum.

3.1. Adding noise

We prepared a dataset of noises that consists of three different sources:

- 240 samples (4 minutes long) taken from the Freesound library[3] (real fan, HVAC, street, city, shop, crowd, library, office and workshop).
- 5 samples (4 minutes long) of artificially generated noises: various spectral modifications of white noise + 50 and 100 Hz hum.
- 18 samples (4 minutes long) of babbling noises by merging speech from 100 random speakers from the Fisher database using a speech activity detector.

Noises were divided into two disjoint groups for training (223 files) and development (40 files).

3.2. Reverberation

We prepared a set of room impulse responses (RIRs) consisting of real room impulse responses from several databases: AIR[4], C4DM[5] (Stewart and Sandler, 2010), MARDY[6], OPENAIR[7], RVB 2014[8], RWCP[9] and RVB 2014[10]. Together, they cover all types of rooms (small rooms, big rooms, lecture room, restrooms, halls, stairs etc.). All room models have more than one impulse response per room (different RIR was used for source of the signal and source of the noise to simulate their different locations). Rooms were split into two disjoint sets, with 396 rooms for training and 40 rooms for development.

3.3. Composition of the training set

To mix the reverberation, noise and signal at a given SNR, we followed the procedure showed in Fig. 3. The pipeline begins with two branches, where speech and noise are reverberated separately. Different RIRs from the same room are used for signal and noise, to simulate different positions of sources.

The next step is A-weighting, applied to simulate the perception of the human ear to added noise (Aarts, 1992). With this filtering, the listener would be able to better perceive the SNR, because most of the noise energy is coming from frequencies that the human ear is sensitive to.

In the following step, we set a ratio of noise and signal energies to obtain the required SNR. Energies of the signal and noise are computed from frames given by original signal’s voice activity detection (VAD). It means the computed SNR is really present in speech frames which are important for SV (frames without voice activity are removed during processing).

The useful signal and noise are then summed at desired SNR, and filtered with telephone channel (see page 9 in ITU, 1994) to compensate for the fact that our noise samples are not coming from the telephone channel, while the original clean data (Fisher) are in fact telephone. The final output is a reverberated and noisy signal with the required SNR, which simulates a recording passing through the telephone channel (as was the original signal) in various acoustic environments. In case we want to add only noise or reverberation, only the appropriate part of the algorithm is used.

4. Experimental setup

4.1. Training data

To train the UBM and the i-vector extractor, we used the PRISM (Ferrer et al., 2011) training dataset definition without added noise or reverberation. The PRISM set comprises Fisher 1 and 2, Switchboard phase 2 and 3 and

[3] http://www.freesound.org
[4] http://www.iks.rwth-aachen.de/en/research/tools-downloads/databases/aachen-impulse-response-database/
[5] http://isophonics.net/content/room-impulse-response-data-set
[6] http://www.commsp.ee.ic.ac.uk/~sap/resources/mardy-multichannel-acoustic-reverberation-database-at-york-database/
[7] http://www.openairlib.net/auralizationdb
[8] http://reverb2014.dereverberation.com/index.html
[9] http://www.openslr.org/13/
[10] http://reverb2014.dereverberation.com/index.html

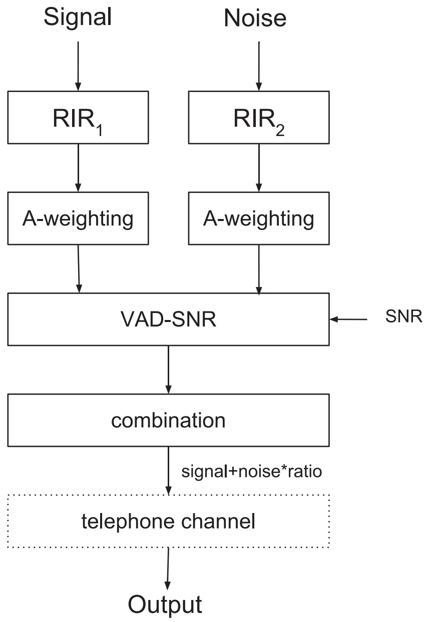


Fig. 3. The process of data augmentation for autoencoder training, generating additional data for PLDA training, or system testing. The last step—filtering with the telephone channel—is used only when creating the denoising autoencoder training data.

Switchboard cellphone phases 1 and 2, along with a set of Mixer speakers. This includes the 66 held out speakers from SRE10 (see Section III-B5 of Ferrer et al., 2011), and 965, 980, 485 and 310 speakers from SRE08, SRE06, SRE05 and SRE04, respectively. A total of 13,916 speakers are available in Fisher data and 1,991 in Switchboard data. Four variants of gender-independent PLDA were trained: the first variant was trained on the clean training data only, while the training sets for the other variants were augmented with artificially added mix of different noises and reverberated data (this portion was based on 30% of the clean training data, i.e. approximately 24k segments, see also Section 4.4).

4.2. Evaluation data

We evaluated our systems on the *female* portions (typically harder subset of trials) of the following NIST SRE 2010 (NIST, 2010) and PRISM conditions:

- *tel-tel*: SRE 2010 extended telephone condition involving normal vocal effort conversational telephone speech in enrollment and test (known as “condition 5”).
- *int-int*: SRE 2010 extended interview condition involving interview speech from different microphones in enrollment and test (known as “condition 2”).
- *int-mic*: SRE 2010 extended interview-microphone condition involving interview enrollment speech and normal vocal effort conversational telephone test speech recorded over a room microphone channel (known as “condition 4”).
- *prism,noi*: Clean and artificially noised waveforms from both interview and telephone conversations recorded over lavalier microphones. Noise was added at different SNR levels and recordings are tested against each other.
- *prism,rev*: Clean and artificially reverberated waveforms from both interview and telephone conversations recorded over lavalier microphones. Reverberation was added with different RTs and recordings are tested against each other.
- *prism,chn*: English telephone conversation with normal vocal effort recorded over different microphones from both SRE2008 and 2010. Recordings are tested against each other.

Additionally, we used the *Core-Core* condition from the SITW challenge − *sitw-core-core*. The SITW (see McLaren et al., 2016) dataset is a large collection of real-world speech from individuals across a wide array of challenging acoustic and environmental conditions. These audio recordings do not contain any artificially added noise, reverberation or other artifacts. This database was collected from open-source media. The *sitw-core-core* condition comprises audio files each containing a continuous speech segment from a single speaker. Enrollment and test segments contain between 6-180 seconds of speech. We evaluated all trials (both genders).

We also tested our systems on the NIST SRE 2016, described in NIST (2016). This is a non-English trial set that we further split by language into Tagalog (*sre16-tgl-f*) and Cantonese (*sre16-yue-f*). We use only female trials (both single- and multi-session). Concerning the experiments with SRE'16, it is important to note that we did not use the SRE'16 unlabeled development set in any way, and we did not perform any score normalization (such as adaptive s-norm).

The speaker verification performance is evaluated in terms of the equal error rate (EER).

4.3. NIST retransmitted set (BUT-RET)

To evaluate the impact of room acoustics on the accuracy of speaker verification, a proper dataset of reverberant audio is needed. An alternative that fills a qualitative gap between unsatisfying simulation (despite the improvement of realism reported in Ravanelli et al., 2016) and costly and demanding real speaker recording, is retransmission. To our advantage, we can also use the fact that a *known* dataset can be retransmitted so that the performances are readily comparable with known benchmarks. Hence, this was the method to obtain a new dataset.

The retransmission took place in a room with floor plan displayed in Fig. 4. The configuration fits several purposes: the loudspeaker−microphone distance rises steadily for microphones 1...6 to study deterioration as a function of distance, microphones 7...12 form a large microphone array mainly focused to explore beamforming (beyond the scope of this paper but studied in Mošner et al., 2018).

For this work, a subset of NIST SRE 2010 data was retransmitted. The dataset consists of 459 female recordings with nominal durations of three and eight minutes. The total number of female speakers is 150. The files were played in sequence and recorded simultaneously by a multi-channel acquisition card that ensured sample precision synchronization.

We denote the retransmitted data as condition *BUT-RET-*∗, where BUT-RET-*orig* represents the original (not retransmitted) data and BUT-RET-*merge* stands for data from 14 conditions: trial scores were produced for all 14 microphones by a single system, these scores were pooled and single EER was evaluated.

4.4. PLDA augmentation sets

For augmenting the PLDA training set, we created new artificially corrupted training sets from the PRISM training data. We used noises and RIRs described in Section 3. To mix the reverberation, noise and signal at given SNR, we followed the procedure outlined in Fig. 3, but omitting the last step of applying the telephone channel. We trained the four following PLDAs (with abbreviations used further in the text):

- **Clean**: PLDA was trained on original PRISM data, without augmentation.
- **N**: PLDA was trained on i) original PRISM data, and ii) a portion (24k segments) of the original training data corrupted by noise.
- **RR**: PLDA was trained on i) original PRISM data, and ii) a portion of the original training data corrupted by reverberation using real room impulse responses.
- **RR+N**: PLDA was trained on i) original PRISM data, ii) noisy augmented data, and iii) reverberated data as described above.

Note that the sizes of all three augmentation sets are the same.

4.5. Augmentation sets for the embedding system

When defining the data set for training the embedding system, we were trying to stay close to the recipe introduced by Snyder (2017), but we introduced modifications to the training data that allowed us to test on a larger set of benchmarks (PRISM, NIST SRE 2010). Every speaker must have at least 6 utterances after augmentation (unlike 8

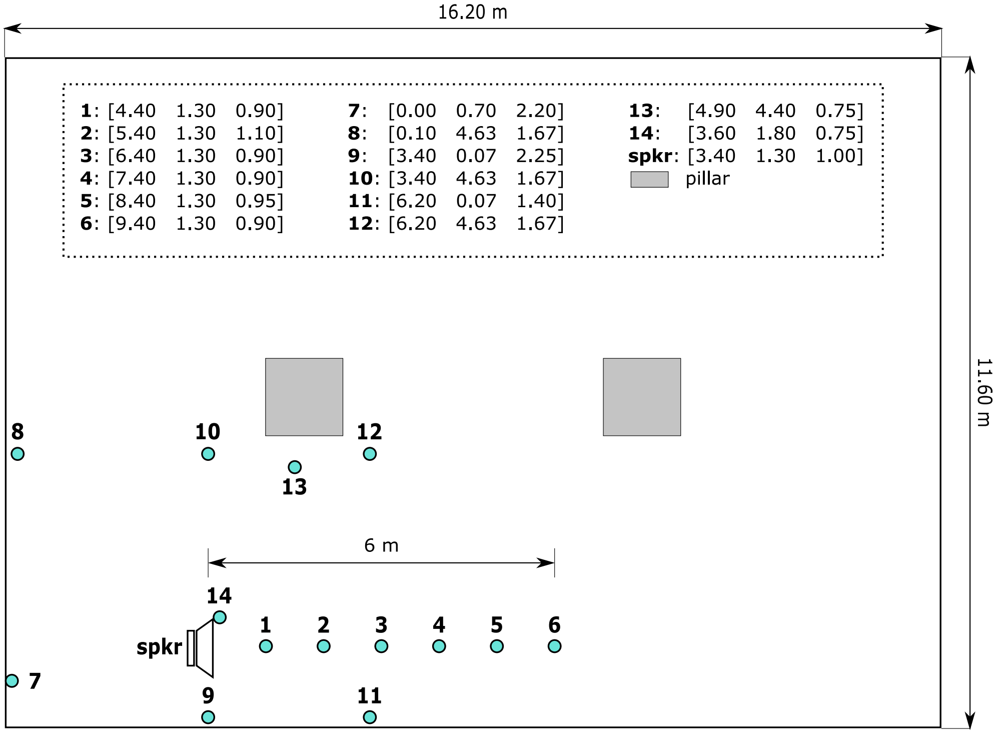


Fig. 4. Floor plan of the room in which the retransmission took place. Coordinates are in meters and lower left corner is the origin.

in the original recipe) and every training sample must be at least 500 frames long. As a consequence of these constraints and given the augmentation process described below, we ended up with 11383 training speakers.

In the original Kaldi recipe, the training data were augmented with reverberation, noise, music, and babble noise and combined with the original clean data. The package of all noises and room impulse responses can be downloaded from OpenSLR[11] (Ko et al., 2017), and includes the MUSAN noise corpus (843 noises).

For data *augmentation with reverberation*, the total amount of RIRs is divided into two equally distributed lists for medium and small rooms.

For *augmentation with noise*, we created three replicas of the original data. The first replica was modified by adding MUSAN noises at SNR levels in the range of 0–15 dB. In this case, the noise was added as a foreground noise (meaning that several non-overlapping noises can be added to the input audio). The second replica was mixed with music as background noise at SNRs ranging from 5 to 15 dB (one noise per audio with the given SNR). The last noisy replica of training data was created by mixing in babble noise. SNR levels were at 13–20 dB and we used 3–7 noises per audio. The augmented data were pooled and a random subset of 200k audios was selected and combined with clean data. The process of data augmentation is also described in Snyder et al. (2018).

Apart from the original recipe, as described in the previous paragraph, we also added our own processing: real room impulse responses and stationary noises described in Section 3. The original RIR list was extended by our list of real RIRs and we kept one reverberated replica. Our stationary noises were used to create another replica of data with SNR levels in the range 0–20 dB. We combined all replicas and selected a subset of 200k files. As a result, after performing all augmentations, we obtain 5 replicas for each original utterance. Finally, we combined all replicas and selected a subset of 200k files. The whole process of creating the x-vector extractor training set is depicted in Fig. 5.

[11] http://www.openslr.org/resources/28/rirs_noises.zip

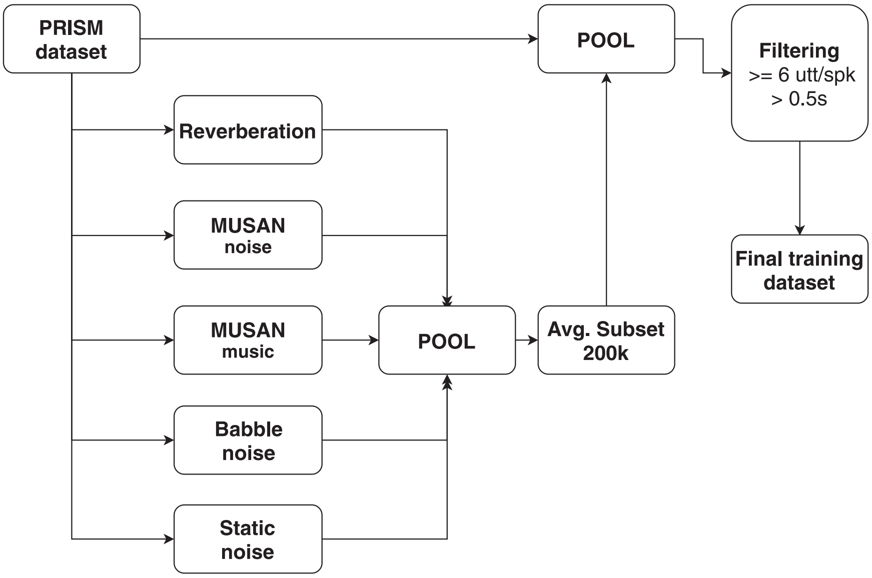


Fig. 5. Preparation of the x-vector extractor training dataset.

5. Experiments and discussion

We provide a set of results, where we study the influence of DNN autoencoder signal enhancement on a variety of systems. Our autoencoder approach is also compared to the multi-condition training of PLDA, which can also improve the performance of the system in corrupted acoustic environments. At the end, we combine the autoencoder with the multi-condition training, and we find this combination beneficial.

We trained autoencoders for signal enhancement, simultaneously for denoising and dereverberation, which provides better robustness towards an unknown form of signal corruption, compared to autoencoders trained on noise or reverberation only (as studied in Novotný et al., 2018a).

We also created different multi-condition training sets for PLDA (described in Section 4.4), similarly as for the autoencoder training (see Section 3). We used exactly the same noises and reverberation for segment corruption as in the autoencoder training, allowing to compare the performance of systems using the autoencoder and systems based on multi-condition training.

Our results are listed in Table 1 for the i-vector-based systems, and in Table 3 for the x-vector based ones. The results in each table are separated into four main blocks based on a combination of features and signal augmentation: i) system trained with MFCC without signal enhancement, ii) system trained with MFCC with signal enhancement, iii) system trained with SBN-MFCC without enhancement, iv) and system trained with SBN-MFCC and signal enhancement. In each block, the first column corresponds to the system where PLDA was trained only on clean data. The next three columns represent results when using different multi-condition training: N, RR or N+RR (as described in Section 4.4).

Finally, the rows of the table are also divided based on the condition, into telephone channel, microphone and artificially created conditions. The last row denoted as *avg* gives the average EER over all conditions and each value set in bold is the minimum EER in the particular condition. We used no domain adaptation nor score normalization that are specific to the NIST SRE16 and required to obtain the best results in this mismatched domain scenario.

5.1. I-vector systems experiments

Let us first compare systems with and without signal enhancement. In this case, we focus on PLDA trained on clean data only. Four scenarios are presented in Table 1. The conditions are divided into four groups based on the nature of the channel—telephone (rows 1−3), microphone (rows 4−7), noisy and reverberated (rows 8−9), and retransmited (rows 10−11), with their average in row 12.

Table 1
Results (EER [%]) obtained in four scenarios. Each block corresponds to an **i-vector** system trained with either MFCC or SBN-MFCC features and with or without signal enhancement applied during i-vector extraction. Blocks are divided into columns corresponding to systems trained in multi-condition fashion (with noised and reverberated data in PLDA). Each column corresponds to a different PLDA multi-condition training set: "—" - clean condition, N - noise, RR - real reverberation, RR+N - real reverberation + noise. The last row denoted as *avg* gives the average EER over all conditions and each value set in bold is the minimum EER in the particular condition.

	MFCC ORIG				MFCC DENOISED				SBN-MFCC ORIG				SBN-MFCC DENOISED			
Condition	—	N	RR	RR+N	—	N	RR	RR+N	—	N	RR	RR+N	—	N	RR	RR+N
1 tel-tel	1.99	2.39	1.99	2.74	2.06	2.48	2.01	2.09	0.94	1.04	0.93	0.93	0.96	0.97	0.94	**0.91**
2 sre16-tgl-f	21.85	21.37	21.84	21.88	23.38	22.96	23.25	23.14	21.88	**21.24**	21.82	21.93	22.62	21.93	22.60	22.70
3 sre16-yue-f	11.20	**10.52**	11.15	11.53	11.76	11.47	11.76	11.79	13.45	13.02	13.45	13.44	14.60	13.69	14.54	14.52
4 int-int	4.57	4.70	4.49	4.55	4.34	4.59	4.21	4.00	3.88	4.07	3.77	3.73	3.44	3.69	**3.40**	**3.40**
5 int-mic	1.85	2.09	1.86	2.00	2.51	2.33	2.40	2.32	1.85	**1.69**	1.76	1.78	1.87	1.79	1.84	1.77
6 prism,chn	1.03	1.29	0.99	0.97	0.59	0.67	0.59	0.57	0.40	0.46	0.39	**0.36**	0.66	0.78	0.72	0.66
7 sitw-core-core	10.11	10.13	10.06	10.32	9.41	9.60	9.45	9.45	8.09	7.85	8.02	8.03	7.71	**7.59**	7.70	7.70
8 prism,noi	3.72	3.02	3.65	3.42	2.51	2.38	2.46	2.38	2.43	1.98	2.45	2.20	1.84	**1.73**	1.81	1.76
9 prism,rev	2.51	2.67	2.40	2.23	1.94	2.09	1.89	1.92	1.42	1.39	1.30	1.31	1.12	1.23	**1.07**	1.09
10 BUT-RET-orig	2.29	2.56	2.30	2.33	2.19	2.48	2.20	2.19	1.45	1.58	1.47	**1.43**	1.82	1.78	1.81	1.80
11 BUT-RET-merge	14.43	14.33	13.79	11.22	11.73	11.51	10.83	10.88	15.27	15.00	15.10	13.32	9.97	10.72	9.38	9.47
12 avg	6.87	6.82	6.77	6.65	6.58	6.60	6.46	6.43	6.46	6.30	6.41	6.22	6.06	5.99	**5.98**	**5.98**

In the first case ("MFCC ORIG" and "MFCC DENOISED" blocks), the i-vector system was trained using MFCC features. In the first set of conditions representing a telephone channel, we generally see a slight degradation when signal enhancement is deployed. Considering that this is a reasonably clean condition, signal enhancement was not expected to be very effective. This is pronounced especially in the *tel-tel* condition, where we have clean English speech, and the degradation is minimal. Our explanation is that, even though the autoencoder was trained also to map clean speech to clean speech in order to cover this scenario, 1) these conditions comprise different telephone channels than the channel used for autoencoder training and, 2) it is likely that the autoencoder introduces a low-level nuisance variability or noise that even correlates with the input channel, and that the speaker modeling technique is incapable of capturing. Slightly higher degradation can be seen in the (non-English) SRE16 telephone conditions, which suggests that the autoencoder is sensitive to language mismatch.

In the second block of conditions (interview speech), the situation is rather different, except for the *int-mic* condition. We can notice an improvement in the system with signal enhancement. An interesting result can be spotted in condition *prism,chn*, where, with signal enhancement, we obtain more than 40 % relative improvement. *prism,chn* is an interview condition with a desk-top microphone, which implies short reverberation in the audio, where the autoencoder can be very effective. This is not so true for the SBN-MFCC system. This confirms our hypothesis that the auto-encoder introduces a low-level noise: although the SBN-MFCC i-vector system is robust to this condition without denoising, it could be sensitive to the potential autoencoder-induced noise.

In the prism and BUT-RET blocks, the situation is very similar, except for the BUT-RET-orig−SBN-MFCC system, which again confirms our hypothesis, as the BUT-RET-orig refers to a clean interview data selection (as described in Section 4.3). We can see significant improvement in the noisy condition *prism,noi* and in the reverberated condition *prism, rev*. This improvement can be further amplified by simultaneously applying denoising and multi-condition PLDA training.

It is worth noting that, looking at the average word error rates in row 10, the auto-encoder enhancement generally improves the system performance.

Let us now focus on the i-vector system based on the SBN-MFCC features. In the past, these features provided good robustness in noisy conditions. We verify this statement comparing columns *MFCC-ORIG* and *SBN-MFCC-ORIG* in Table 1 (systems without signal enhancement). We see that, except for the SRE 2016 and *BUT-RET-merge* conditions, the systems trained with stacked bottle-neck features yield better performance compared to the corresponding original MFCC systems. In case of degradation of the (non-English) SRE 2016, we are dealing with similar situation as described in MFCC i-vector systems. The bottle-neck network is trained on English data, so we have to deal with different language in bottle-neck training, which can cause this degradation. In general, these results suggest that SBN training and usage is sensitive to specific domains.

When comparing systems with and without signal enhancement, the situation is similar to the MFCC case. We see degradation on the telephone channels and a portion of the interview speech conditions. We obtain 30 % relative improvement in *BUT-RET-merge* where the system without enhancement is even worse than the previous i-vector system. This could indicate that the bottle-neck features provide better robustness to noise than to reverberation.

Let us note that each system has been evaluated four times on each condition, based on what data we used in the PLDA training: "—" - clean condition, N - noise, RR - real reverberation, RR+N - real reverberation + noise. It can be generally stated, that, according to our expectations, adding the corresponding domain data (e.g. reverberated data for reverberated condition) is beneficial. However, when looking at the average results, adding both reverberated and noisy data to the PLDA training generally helps.

In Section 4.5, we described the augmentation setup for the x-vector system compared to the i-vector extractor training setup. Our presented i-vector extractors were trained on the original clean data only. Our hypothesis is that generative i-vector extractor training does not benefit from data augmentation as much as the x-vectors do. The comparison of our MFCC i-vector extractor trained on the original clean data and augmented data (the type of augmentation is the same as described in Section 3) is shown in Table 2. We see some improvement in some conditions, but mostly degradation. The reason is that generative i-vector extraction training is unsupervised. When we add augmented data to the training list, i-vector extractor is forced to reserve a portion of its parameters for representing the variability caused by noise and reverberation, and so it limits the number of parameters available for modeling speaker variability. In the supervised discriminative x-vector approach, we are forcing the x-vector extractor to do the opposite. The extractor is forced to distinguish the speakers, and data augmentation in the training can be beneficial.

5.2. X-vector systems experiments

We evaluated our speech enhancement autoencoder also with the system based on x-vectors, that are currently considered as the state-of-the-art speaker verification technique. In our experiments and system design, we have deviated from the original Kaldi recipe (Snyder et al., 2018). For training the x-vector extractor, we extended the number of speakers and we also created more variants of augmented data. We extended the original data augmentation recipe by adding real room impulse responses and an additional set of stationary noises (the extension process is also described in Novotný et al. (2018b), the x-vector network used here is labeled as Aug III. in the paper). In the PLDA back-end training, we also added the augmented data for multi-condition training (see Section 4.4).

Let us point out that the signal enhancement autoencoder was trained on a subset of augmented data for training the x-vector DNN. The set of noises and real room impulse responses are therefore the same as in our extended set for training the x-vector extractor (as described in Section 3) and there is no advantage in providing the autoencoder

Table 2
Results (EER [%]) of i-vector extractor trained on clean data (*iX ORIG*) compared to i-vector extractor trained on augmented data (*iX AUG*). Blocks are divided into columns corresponding to systems trained in multi-condition fashion (with noised and reverberated data in PLDA). Each column corresponds to a different PLDA multi-condition training set: "—" - clean condition, N - noise, RR - real reverberation, RR+N - real reverberation + noise.

	iX ORIG				iX AUG			
Condition	—	N	RR	RR+N	—	N	RR	RR+N
tel-tel	1.99	2.39	1.99	2.74	**1.98**	2.44	1.96	2.86
sre16-tgl-f	21.85	**21.37**	21.84	21.88	22.33	21.95	22.06	22.62
sre16-yue-f	11.20	**10.52**	11.15	11.53	11.32	10.59	11.26	11.20
int-int	4.57	4.70	4.49	4.55	4.52	4.88	**4.44**	4.71
int-mic	**1.85**	2.09	1.86	2.00	2.11	2.17	2.04	2.02
prism,chn	1.03	1.29	0.99	0.97	0.92	1.20	0.95	1.04
sitw-core-core	10.11	10.13	**10.06**	10.32	10.28	10.38	10.17	10.34
prism,noi	3.72	**3.02**	3.65	3.42	3.79	3.03	3.73	3.26
prism,rev	2.51	2.67	2.40	2.23	2.74	2.80	2.55	**2.22**
BUT-RET-orig	**2.29**	2.56	2.30	2.33	2.56	2.68	2.47	2.64
BUT-RET-merge	14.43	14.33	13.79	11.22	11.16	11.08	10.80	**9.06**

with additional augmentations. It is also useful to refer the interested reader to our analysis in Novotný et al. (2018b), where we show the benefit of having such a large augmentation set for x-vector extractor training.

Let us first compare the x-vector network trained with original MFCC and with SBN-MFCC features. As shown in Table 1 and in Section 5.1, in systems based on i-vectors, bottle-neck features often provided significant improvement, but for x-vector-based systems, the gains are either much lower or the performance stays the same or even degrades for condition *BUR-RET-merge*. This degradation, however, completely disappears after using signal enhancement in x-vector training and subsequently multi-condition training of PLDA. For the telephone data with low reverberation, we can observe either steady performance on *tel-tel* or slightly better performance on more challenging conditions and non-English data in SRE’16. This is in contrast with i-vectors, where we only see either steady performance on easy *tel-tel* or degradation on more challenging SRE’16. In general, the positive effect of SBN-MFCC features on the x-vector system is small, but more stable than on the i-vector system.

When we focus on the effect of signal enhancement in the x-vector-based system, we see much higher improvement compared to i-vectors. There are still several cases where the enhancement causes degradation (MFCC: *int-mic, BUT-RET-orig*; SBN-MFCC: *tel-tel, int-mic, BUT-RET-orig*—generally clean conditions). Otherwise, the enhancement provides a significant improvement across the rest of the systems.

At this point, it is useful to point out that unlike with i-vectors, where signal enhancement is applied only for i-vector extraction, we actually apply enhancement already on top of x-vector training data. The effect of applying enhancement only during x-vector extraction (like with i-vectors) can be seen in Table 4. We can observe that also here, we gain some improvements, but they are generally smaller than with enhancement deployed already during x-vector training (which can be observed in Table 3).

X-vector systems generally provide greater robustness across different signal corruptions. It was natural for us to expect that x-vector systems should not need signal enhancement, and that they would implicitly learn it themselves, especially in the first part of DNN described in Section 2.3. To our belief, a reason why enhancement helped in our case, is that enhancement is not the target task of the x-vector DNN. Even though we did have multiple corrupted samples per speaker in the DNN training set, it may be possible that we simply didn’t have enough. And since the x-vector training is generally known to be data-hungry, it is therefore likely that if we had more corrupted samples per speaker, it would be in the DNN’s natural capabilities to learn the task of de-noising.

Let us also point out that if a single type of noise (or channel in general) appears systematically with a given speaker, the noise becomes a part of the speaker identity and therefore the NN does not compensate for it.

So far, we have compared results on systems where PLDA was trained on clean data only and we have studied possible improvements across several systems. Multi-condition training of PLDA, where we add a portion of augmented data into PLDA training is another possible approach on how to improve system performance and its robustness.

Table 3
Results (EER [%]) obtained in four scenarios. Each block corresponds to an **x-vector** system trained with different type of features with or without signal enhancement. Blocks are divided into columns corresponding to systems trained in multi-condition fashion (with noised and reverberated data in PLDA). Each column corresponds to different PLDA multi-condition training set: “—” - clean condition, N - noise, RR - real reverberation, RR+N - real reverberation + noise. The last row denoted as *avg* gives the average EER over all conditions and each value set in bold is the minimum EER in the particular condition.

	MFCC ORIG				MFCC DENOISED				SBN-MFCC ORIG				SBN-MFCC DENOISED			
Condition	—	N	RR	RR+N	—	N	RR	RR+N	—	N	RR	RR+N	—	N	RR	RR+N
tel-tel	1.30	1.43	1.27	1.29	1.21	1.44	**1.18**	1.20	1.30	1.49	1.29	1.27	1.35	1.45	1.30	1.30
sre16-tgl-f	22.73	22.52	22.87	22.56	21.52	21.41	21.29	21.31	22.33	21.21	22.15	22.33	21.17	**20.74**	20.88	20.95
sre16-yue-f	10.36	9.61	10.45	10.61	8.86	**8.23**	8.75	8.66	9.60	8.71	9.56	9.88	8.89	8.38	8.67	8.64
int-int	3.36	3.72	3.29	3.22	2.92	3.34	2.90	**2.86**	3.24	3.66	3.16	3.20	3.16	3.42	3.08	2.97
int-mic	1.33	1.43	1.3	1.22	1.47	1.37	1.41	1.37	1.07	1.17	1.04	**1.03**	1.37	1.39	1.29	1.27
prism,chn	0.62	0.81	0.61	0.61	0.37	**0.27**	0.36	0.41	0.69	0.67	0.67	0.59	0.36	0.41	0.36	0.36
sitw-core-core	7.87	7.30	7.72	7.41	6.81	6.54	6.73	6.70	7.57	7.42	7.57	7.38	6.81	**6.40**	6.71	6.66
prism,noi	2.76	1.90	2.63	2.11	1.84	**1.50**	1.80	1.73	2.72	1.97	2.63	2.22	1.84	1.57	1.81	1.68
prism,rev	2.08	2.02	1.79	1.60	1.16	1.13	**1.10**	1.12	1.98	2.06	1.71	1.59	1.24	1.27	1.15	1.15
BUT-RET-orig	1.73	1.73	1.69	1.63	1.81	1.82	1.72	1.74	1.50	1.63	1.46	**1.43**	1.77	1.86	1.74	1.75
BUT-RET-merge	15.48	13.94	13.96	13.12	11.83	12.81	**10.07**	10.46	17.20	14.09	15.90	13.74	13.26	12.70	11.03	10.12
avg	6.33	6.04	6.14	5.94	5.44	5.44	5.21	5.23	6.29	5.83	6.10	5.88	5.57	5.42	5.27	**5.17**

Table 4
Results (EER [%]) of SV systems (MFCC or SBN-MFCC as the features) with x-vector extractor trained on original data and with signal enhancement used only for x-vector extraction. Blocks are divided into columns corresponding to systems trained in multi-condition fashion (with noised and reverberated data in PLDA). Each column corresponds to a different PLDA multi-condition training set: "—" - clean condition, N - noise, RR - real reverberation, RR+N - real reverberation + noise.

	MFCC				SBN-MFCC			
Condition	—	N	RR	RR+N	—	N	RR	RR+N
tel-tel	1.38	1.51	1.34	1.39	1.27	1.40	**1.21**	1.25
sre16-tgl-f	21.12	21.48	21.08	**20.94**	21.73	21.46	21.50	21.63
sre16-yue-f	9.76	**9.01**	9.70	9.69	9.38	9.07	9.41	9.16
int-int	3.15	3.32	3.12	**2.99**	3.19	3.40	3.14	3.05
int-mic	1.61	1.67	1.59	1.58	1.63	1.58	1.51	**1.39**
prism,chn	0.54	0.47	0.55	0.54	**0.40**	0.41	**0.40**	**0.40**
sitw-core-core	7.22	6.76	7.17	6.84	6.96	**6.52**	6.97	6.76
prism,noi	2.14	**1.64**	2.15	2.05	2.33	1.67	2.36	2.15
prism,rev	1.24	1.22	1.18	**1.20**	1.33	1.45	1.28	1.24
BUT-RET-orig	**1.87**	2.07	1.90	1.88	2.09	2.03	2.08	2.07
BUT-RET-merge	12.76	11.76	**10.71**	11.83	15.08	14.32	12.62	12.66

From the results, we can see that multi-condition training can provide improvement across all conditions and systems without signal enhancement. We can see that the ideal combination of the augmented data for multi-condition training of PLDA depends on the condition. In noisy condition (*prism,noi*), it is more effective to use noise augmentation only. For reverberated condition (*prism,rev, BUT-RET-merge*) we can see more benefits in using the reverberated augmentation set compared to others.

5.3. Final remarks

Although EER is a common metric summarizing performance, it does not cover all operating points. In this section, we present the performance of various systems via DET and DCF curves as to see the behaviour of systems in a more comprehensive way.

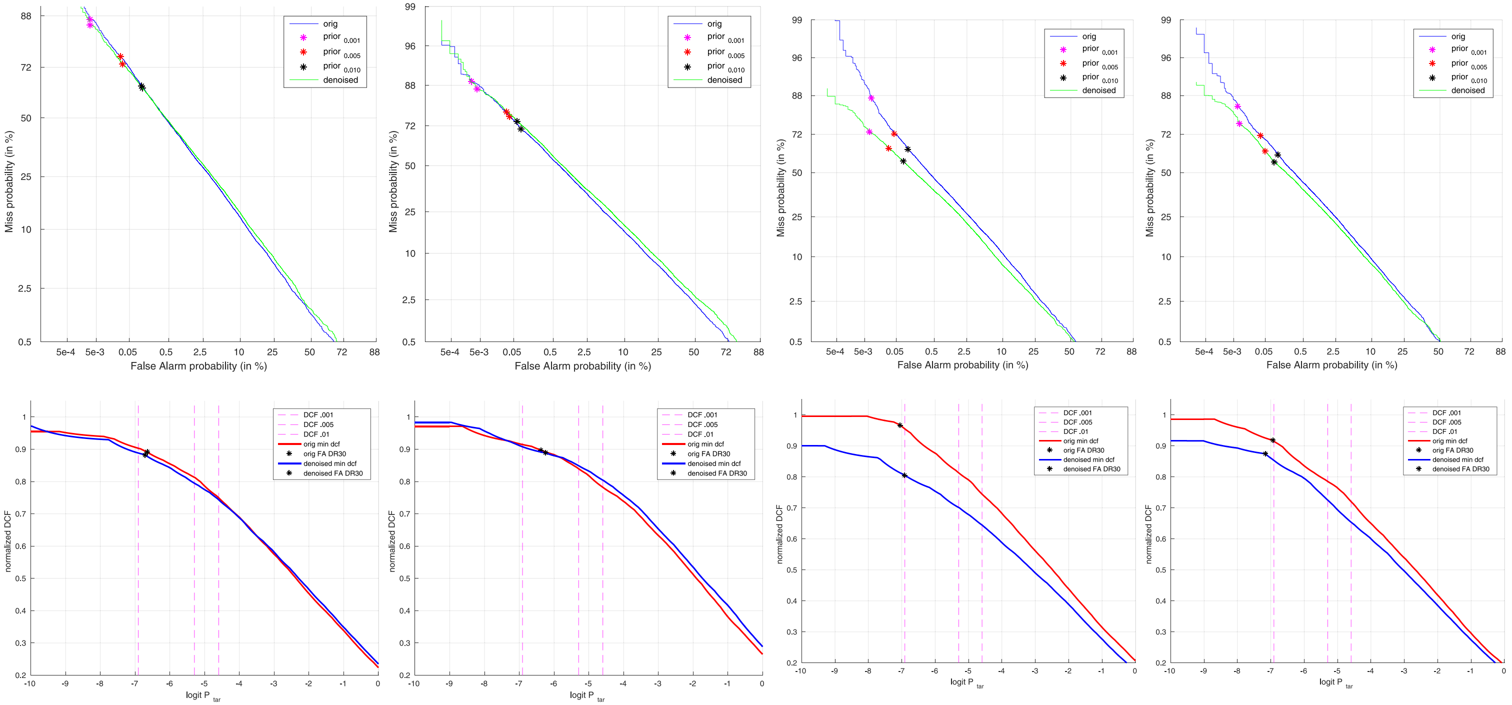
In order to summarize our observation without overwhelming the reader with too many plots, we have chosen two representative conditions, that are the closest to real-world scenario—*sre16-yue-f* (Fig. 6) and *BUT-RET-merge* (Fig. 7). More specifically, the *sre16-yue-f* condition was chosen because a) it contains original noisy audio, and b) compared to the rest of the conditions, there is a high channel mismatch between the training data and the evaluation data. The *BUT-RET-merge* condition was chosen because it realistically reflects real reverberation.

Looking at the graphs reveals that the benefit from using the studied techniques can be substantial. It is worth noting that according to the tables above, signal enhancement may not be effective in terms of EER. However, when looking at the DET curves, we see that there are operating points that do benefit from denoising to a fairly large extent. Otherwise, it can seem that signal enhancement improvement vanishes in some operation points, compared to the significant improvement in EER: for example, BUT-merge, where improvement from speech enhancement is vanishing in the upper-left region. It is important to know which operation point is in our focus. In the selected upper-left region, the false-alarm probability is too small for any conclusion.

Looking at the DET or DCF curves covering a wide range of operating points, we can observe that signal enhancement often brings a substantial improvement and in the worst case it does not deteriorate the overall performance (e.g. *sre16-yue-f* condition).

6. Conclusion

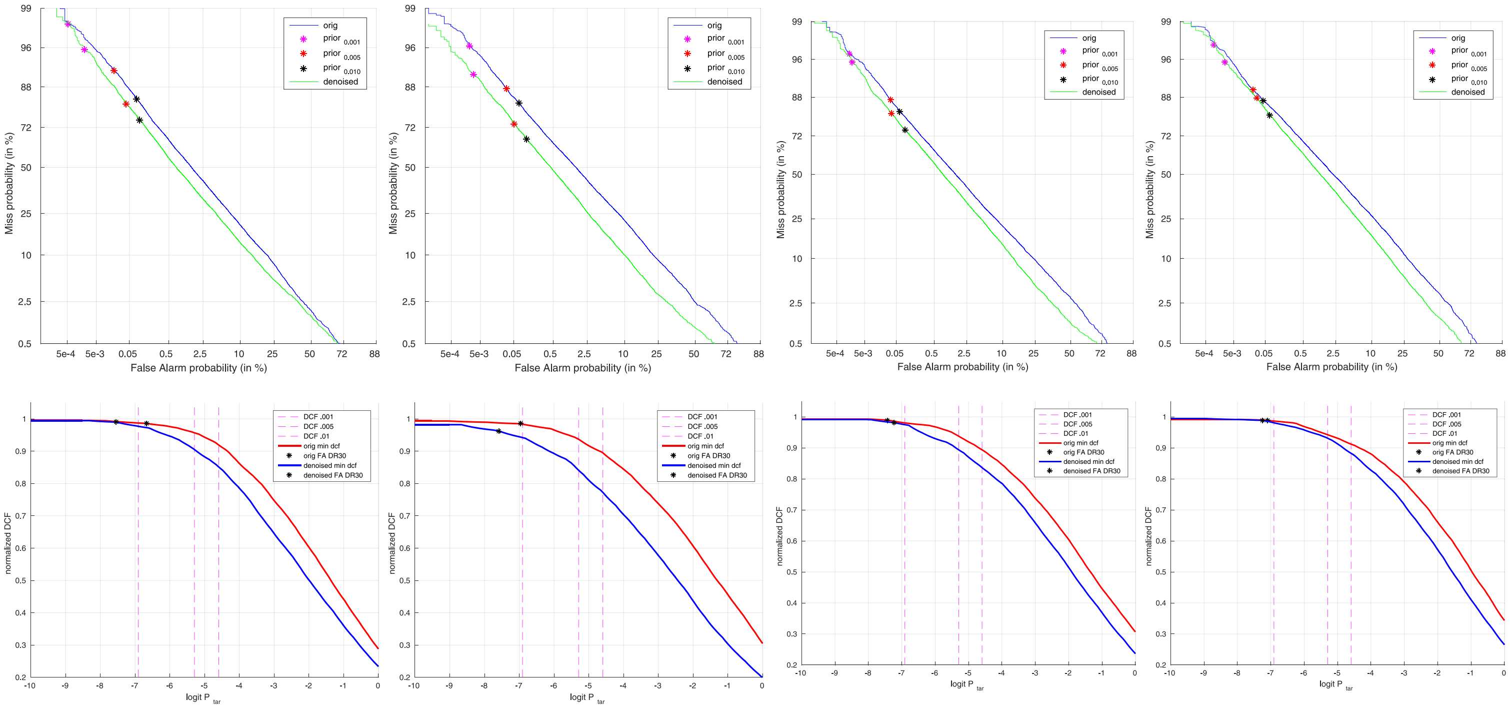
In this paper, we have analyzed several aspects of DNN-autoencoder enhancement for designing robust SV systems. We have studied the influence of signal enhancement on different speaker verification system paradigms (generative i-vectors vs. discriminative x-vectors) and we have analyzed possible improvements with different features.



(a) I-vector based systems: left column MFCC features, right column SBN-MFCC features.

(b) X-vector based systems: left column MFCC features, right column SBN-MFCC features.

Fig. 6. Detection Error Trade-off (DET) plots (top row) and minDFC as a function of effective prior (bottom row) of all tested scenarios for *sre16-yue-f* condition. Intersection of minDCF curves with vertical dashed violet lines correspond (from left to right) to the minDCF from NIST SRE 2010 and to the two operating points of DCF from NIST SRE2016. Similarly, the violet star in the DET plots corresponds to the minDCF from NIST SRE2010 and red and black stars correspond to the two operating points of the NIST SRE 2016.



(a) I-vector based systems: left column MFCC features, right column SBN-MFCC features.

(b) X-vector based systems: left column MFCC features, right column SBN-MFCC features.

Fig. 7. Detection Error Trade-off (DET) plots (top row) and minDFC as a function of effective prior (bottom row) of all tested scenarios for *BUT-RET-merge* condition. Intersection of minDCF curves with vertical dashed violet lines correspond (from left to right) to the minDCF from NIST SRE 2010 and to the two operating points of DCF from NIST SRE2016. Similarly, the violet star in the DET plots corresponds to the minDCF from NIST SRE2010 and red and black stars correspond to the two operating points of the NIST SRE 2016.

Our results suggest that the DNN autoencoder speech signal enhancement can be helpful to improve system robustness against noise and reverberation without the risk of significant degradation under clean conditions, making it a stable and universal technique for robustness improvement that is independent on the system. We have also compared the PLDA multi-condition training with audio enhancement. Both approaches are complementary and the systems can benefit from simultaneous usage of both.

The results under clean conditions (tel-tel, sre16-*) show a weak spot in the usage of the autoencoder. In general, it is likely that the autoencoder introduces a low-level nuisance variability or noise that is in effect lower than—and diminishes in—the original variability of interest (reverberation, noise), however, the underlying model is incapable of capturing it. Also, the training data for the autoencoder come from a different domain than the test data. For the sre16-* conditions—being non-English—we further suspect that the degradation in performance (mainly) due to language mismatch. This gives us motivation for future research in language dependency of signal enhancement.

Our future research will focus on incorporating signal enhancement directly in the x-vector extractor by the means of multi-task training, combining speaker separation and signal enhancement objective functions and possibly taking even more benefit from their joint optimization.

Acknowlgedgments

The work was supported by Czech Ministry of Interior project No. VI20152020025 “DRAPAK, Google Faculty Research Award program, Czech Science Foundation (GACR) project No. GJ17-23870Y, Czech National Science Foundation (GACR) project “NEUREM3 No. 19-26934X, and by Czech Ministry of Education, Youth and Sports from the National Programme of Sustainability (NPU II) project “IT4Innovations excellence in science - LQ1602.

References

Aarts, R.M., 1992. A comparison of Some Loudness Measures for Loudspeaker Listening Tests. J. Audio Eng. Soc 40 (3), 142–146.http://www.extra.research.philips.com/hera/people/aarts/RMA_papers/aar92a.pdf.

Bhattacharya, G., Alam, J., Kenny, P., 2017. Deep Speaker Embeddings for Short-Duration Speaker Verification. Interspeech 2017, pp. 1517–1521.

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

Dehak, N., Kenny, P., Dehak, R., Dumouchel, P., Ouellet, P., 2011. Front-End Factor Analysis For Speaker Verification. IEEE Transactions on Audio, Speech, and Language Processing 19 (4), 788–798. doi: 10.1109/TASL.2010.2064307.

Dufera, B., Shimamura, T., 2009. Reverberated speech enhancement using neural networks. In: Proc. International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2009., pp. 441–444.

ETSI, 2007. Speech Processing, Transmission and Quality Aspects (STQ). Technical Report. European Telecommunications Standards Institute (ETSI). ETSI ES 202 050.

Ferrer, L., Bratt, H., Burget, L., Cernocky, H., Glembek, O., Graciarena, M., Lawson, A., Lei, Y., Matejka, P., Plchot, O., Scheffer, N., 2011. Promoting robustness for speaker modeling in the community: the PRISM evaluation set. In: Proceedings of SRE11 analysis workshop. Atlanta.

Garcia-Romero, D., Espy-Wilson, C.Y., 2011. Analysis of i-vector length normalization in Gaussian-PLDA speaker recognition systems. In: Proc. Interspeech.

Ghahabi, O., Hernando, J., 2014. Deep belief networks for i-vector based speaker recognition. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1700–1704. doi: 10.1109/ICASSP.2014.6853888.

Glembek, O., Ma, J., Matějka, P., Zhang, B., Plchot, O., Burget, L., Matsoukas, S., 2014. Domain Adaptation via Within-class Covariance Correction in I-Vector Based Speaker Recognition Systerms. In: Proceedings of ICASSP 2014, pp. 4060–4064.

Heigold, G., Moreno, I., Bengio, S., Shazeer, N., 2016. End-to-end text-dependent speaker verification. 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5115–5119. doi: 10.1109/ICASSP.2016.7472652.

ITU, 1994. ITU-T O.41. https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-O.41-199410-I!!PDF-E&type=items.

Karafiát, M., Grézl, F., Veselý, K., Hannemann, M., Szőke, I., Černocký, J., 2014. BUT 2014 Babel system: Analysis of adaptation in NN based systems. Interspeech 2014, pp. 3002–3006.

Kenny, P., 2010. Bayesian speaker verification with Heavy−Tailed Priors. keynote presentation, Proc. of Odyssey 2010.

Ko, T., Peddinti, V., Povey, D., Seltzer, M.L., Khudanpur, S., 2017. A study on data augmentation of reverberant speech for robust speech recognition. 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5220–5224. doi: 10.1109/ICASSP.2017.7953152.

Kumatani, K., Arakawa, T., Yamamoto, K., McDonough, J., Raj, B., Singh, R., Tashev, I., 2012. Microphone Array Processing for Distant Speech Recognition: Towards Real-World Deployment. APSIPA Annual Summit and Conference. Hollywood, CA, USA.

Laskowski, K., Edlund, J., 2010. A Snack implementation and Tcl/Tk Interface to the Fundamental Frequency Variation Spectrum Algorithm. In: Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC’10). Valletta, Malta.

Lei, Y., Burget, L., Ferrer, L., Graciarena, M., Scheffer, N., 2012. Towards Noise-Robust Speaker Recognition Using Probabilistic Linear Discriminant Analysis. In: Proceedings of ICASSP. Kyoto, JP.

Lei, Y., Scheffer, N., Ferrer, L., McLaren, M., 2014. A novel scheme for speaker recognition using a phonetically-aware deep neural network. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 1695–1699. doi: 10.1109/ICASSP.2014.6853887.

Lozano-Diez, A., Silnova, A., Matějka, P., Glembek, O., Plchot, O., Pešán, J., Burget, L., Gonzalez-Rodriguez, J., 2016. Analysis and Optimization of Bottleneck Features for Speaker Recognition. In: Proceedings of Odyssey 2016, 2016, pp. 352–357.

Martínez, D.G., Burget, L., Stafylakis, T., Lei, Y., Kenny, P., LLeida, E., 2014. Unscented Transform For Ivector-based Noisy Speaker Recognition. In: Proceedings of ICASSP 2014. Florencie, Italy.

Matějka, P., Glembek, O., Novotný, O., Plchot, O., Grézl, F., Burget, L., Černocký, J., 2016. Analysis Of DNN Approaches To Speaker Identification. In: Proceedings of the 41th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016), 2016, pp. 5100–5104.

Matějka, P., Novotný, O., Plchot, O., Burget, L., Diez, M.S., Černocký, J., 2017. Analysis of Score Normalization in Multilingual Speaker Recognition. In: Proceedings of Interspeech 2017, 2017, pp. 1567–1571. doi: 10.21437/Interspeech.2017-803.

Matějka, P., Burget, L., Schwarz, P., Černocký, J., 2006. Brno University of Technology System for NIST 2005 Language Recognition Evaluation. In: Proceedings of Odyssey 2006. San Juan, Puerto Rico.

Matějka, P., et al., 2014. Neural network bottleneck features for language identification. IEEE Odyssey: The Speaker and Language Recognition Workshop. Joensu, Finland.

McLaren, M., Ferrer, L., Castan, D., Lawson, A., 2016. The Speakers in the Wild (SITW) Speaker Recognition Database. Interspeech 2016, pp. 818–822. doi: 10.21437/Interspeech.2016-1129.

Mimura, M., Sakai, S., Kawahara, T., 2014. Reverberant speech recognition combining deep neural networks and deep autoencoders. In: Proc. Reverb Challenge Workshop. Florence, Italy.

Mošner, L., Matějka, P., Novotný, O., Černocký, J., 2018. Dereverberation and beamforming in far-field speaker recognition. In: Proceedings of ICASSP.

NIST, 2010. The NIST year 2010 Speaker Recognition Evaluation Plan. https://www.nist.gov/sites/default/files/documents/itl/iad/mig/NIST_SRE10_evalplan-r6.pdf.

NIST, 2016. The NIST year 2016 Speaker Recognition Evaluation Plan. https://www.nist.gov/sites/default/files/documents/2016/10/ 07/sre16_eval_plan_v1.3.pdf.

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

Novotný, O., Matějka, P., Plchot, O., Glembek, O., 2018. On the use of DNN Autoencoder for Robust Speaker Recognition. Technical Report.

Novotný, O., Matvějka, P., Glembek, O., Plchot, O., Grézl, F., Burget, L., Černocký, J., 2016. Analysis of the DNN-based SRE systems in multi-language conditions. 2016 IEEE Spoken Language Technology Workshop (SLT), pp. 199–204. doi: 10.1109/SLT.2016.7846265.

Novotný, O., Plchot, O., Matějka, P., Mošner, L., Glembek, O., 2018. On the use of X-vectors for Robust Speaker Recognition. In: Proceedings of Odyssey 2018, 2018, pp. 168–175. doi: 10.21437/Odyssey.2018-24.

Peddinti, V., Povey, D., Khudanpur, S., 2015. A time delay neural network architecture for efficient modeling of long temporal contexts. INTERSPEECH 2015, 16th Annual Conference of the International Speech Communication Association, Dresden, Germany, September 6-10, 2015, pp. 3214–3218.

Pelecanos, J., Sridharan, S., 2006. Feature Warping for Robust Speaker Verification. In: Proceedings of Odyssey 2006: The Speaker and Language Recognition Workshop. Crete, Greece.

Plchot, O., Burget, L., Aronowitz, H., Matějka, P., 2016. Audio Enhancing With DNN Autoencoder For Speaker Recognition. In: Proceedings of the 41th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP 2016), 2016, pp. 5090–5094.

Plchot, O., Matsoukas, S., Matějka, P., Dehak, N., Ma, J., Cumani, S., Glembek, O., Heřmanský, H., Mesgarani, N., Soufifar, M.M., Thomas, S., Zhang, B., Zhou, X., 2013. Developing A Speaker Identification System For The DARPA RATS Project. In: Proceedings of ICASSP 2013, Vancouver, CA.

Plchot, O., Matvějka, P., Silnova, A., Novotný, O., Sánchez, M.D., Rohdin, J., Glembek, O., Brümmer, N., Swart, A., JorrÃn-Prieto, J., García, P., Buera, L., Kenny, P., Alam, J., Bhattacharya, G., 2017. Analysis and Description of ABC submission to NIST SRE 2016. In: Proc. Interspeech 2017, pp. 1348–1352. doi: 10.21437/Interspeech.2017-1498.

Prince, S.J.D., 2007. Probabilistic linear discriminant analysis for inferences about identity. In: Proc. International Conference on Computer Vision (ICCV). Rio de Janeiro, Brazil.

Ravanelli, M., Svaizer, P., Omologo, M., 2016. Realistic Multi-Microphone Data Simulation for Distant Speech Recognition. Interspeech 2016, pp. 2786–2790. doi: 10.21437/Interspeech.2016-731.

Rohdin, J., Silnova, A., Diez, M., Plchot, O., Matějka, P., Burget, L., 2018. End-to-end DNN based speaker recognition inspired by i-vector and PLDA. In: Proceedings of ICASSP.

Snyder, D., 2017. NIST SRE 2016 Xvector Recipe. https://david-ryan-snyder.github.io/2017/10/04/model_sre16_v2.html.

Snyder, D., Garcia-Romero, D., Povey, D., Khudanpur, S., 2017. Deep Neural Network Embeddings for Text-Independent Speaker Verification. Proc. Interspeech 2017 999–1003.

Snyder, D., Garcia-Romero, D., Sell, G., Povey, D., Khudanpur, S., 2018. X-vectors: Robust DNN Embeddings for Speaker Recognition. In: Proceedings of ICASSP.

Snyder, D., Ghahremani, P., Povey, D., Garcia-Romero, D., Carmiel, Y., Khudanpur, S., 2016. Deep neural network-based speaker embeddings for end-to-end speaker verification. 2016 IEEE Spoken Language Technology Workshop (SLT), pp. 165–170. doi: 10.1109/SLT.2016.7846260.

Stewart, R., Sandler, M., 2010. Database of omnidirectional and B-format room impulse responses. 2010 IEEE International Conference on Acoustics, Speech and Signal Processing, pp. 165–168. doi: 10.1109/ICASSP.2010.5496083.

Sturim, D.E., Reynolds, D.A., 2005. Speaker adaptive cohort selection for Tnorm in text-independent speaker verification. In: Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005., 1, pp. I/741–I/744 Vol. 1. doi: 10.1109/ICASSP.2005.1415220.

Swart, A., Brümmer, N., 2017. A Generative Model for Score Normalization in Speaker Recognition. Interspeech 2017, 18th Annual Conference of the International Speech Communication Association, Stockholm, Sweden, August 20-24, 2017, pp. 1477–1481.

Talkin, D., 1995. A Robust Algorithm for Pitch Tracking (RAPT). In: Kleijn, W.B., Paliwal, K. (Eds.), Speech Coding and Synthesis. Elsevier, New York.

Variani, E., Lei, X., McDermott, E., Moreno, I.L., Gonzalez-Dominguez, J., 2014. Deep neural networks for small footprint text-dependent speaker verification. 2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4052–4056. doi: 10.1109/ICASSP.2014.6854363.

Xu, Y., Du, J., Dai, L.-R., Lee, C.-H., 2014. An Experimental Study on Speech Enhancement Based on Deep Neural Networks. IEEE Signal processing letters 21 (1).

Xu, Y., Du, J., Dai, L.-R., Lee, C.-H., 2014. Global variance equalization for improving deep neural network based speech enhancement. In: Proc. IEEE China Summit & International Conference on Signal and Information Processing (ChinaSIP), pp. 71–75.

Yanhui, T., Jun, D., Yong, X., Lirong, D., Chin-Hui, L., 2014. Deep Neural Network Based Speech Separation for Robust Speech Recognition. In: Proceedings of ICSP2014, pp. 532–536.

Zhang, S.X., Chen, Z., Zhao, Y., Li, J., Gong, Y., 2016. End-to-End attention based text-dependent speaker verification. 2016 IEEE Spoken Language Technology Workshop (SLT), pp. 171–178. doi: 10.1109/SLT.2016.7846261.