BRNO UNIVERSITY OF TECHNOLOGY FACULTY OF INFORMATION TECHNOLOGY

Excel@FIT 2019
http://excel.fit.vutbr.cz

Robust Speaker Verification Using Deep Neural Networks

Ján Profant*

Abstract
The objective of this work is to study state-of-the-art deep neural networks based speaker verification systems called x-vectors on wideband conditions, such as YouTube. This system takes variable length audio recording and maps it into fixed length embedding which is afterward used to represent the speaker. We compared our systems to BUT's submission to Speakers in the Wild Speaker Recognition Challenge (SITW). We observed, that when comparing single best systems, with recently published x-vectors we were able to obtain more than 4.38 times lower Equal Error Rate on SITW core-core condition compared to SITW submission from BUT. Moreover, we find that diarization substantially reduces error rate when there are multiple speakers for SITW core-multi condition but we could not see the same trend on NIST Speaker Recognition Evaluation 2018 Video Annotations for YouTube data.

Keywords: speaker verification, neural networks, x-vector

Supplementary Material:

*xprofa00@stud.fit.vutbr.cz, *Faculty of Information Technology, Brno University of Technology*

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. Similarly to the computer vision face recognition, embedding this information into fixed length vector is used.

Using deep neural networks for a topic of speaker verification shows as a very active area of research in the last years [1, 2, 3]. In this approach, time-delay neural network which works on frame level is used and during training, it is trained to classify large dataset of speakers. Long-term speaker characteristics are captured in the network by a temporal pooling layer that aggregates over the input speech. Eventually, fixed-dimensional embeddings from the layer in a network after frame level are used to represent speaker utterance and these are called x-vectors. These embeddings might be scored using euclidean distance, cosine distance but more common is to use backend with Probability Linear Discriminant Analysis (PLDA). In this paper we experiment with this state-of-the-art technique, we compare it to previously used i-vectors [4]. The standard i-vector approach consists of a universal background model (UBM), and a large projection matrix T, that are learned in an unsupervised way to maximize the data likelihood. The projection maps high-dimensional statistics from the UBM into a low-dimensional representation, known as an i-vector. The DNNs most often found in speaker recognition are trained as acoustic models for automatic speech recognition and are then used to enhance phonetic modeling in the i-vector. In recent years, i-vectors started to be replaced by feed-forward neural networks, because of better performance and also because of the wide use of graphical computing units. In this paper, we analyze the performance of both approaches and we focus on using deep neural networks for speaker verification.

We also introduce numerous modifications to Kaldi [5] recipe [3], which was publicly released for the research community. We also summarized our effort during NIST Speaker Recognition Evaluation

2018 where one of our systems was used for final submission in wideband condition.

2. Theoretical Background

2.1 Speaker Recognition

Speaker recognition is the identification of a person from characteristics of voices. No two individuals sound identical because their vocal tract shapes, larynx sizes, and other parts of their voice production organs are different. In addition to these physical differences, each speaker has characteristic manner of speaking, including the use of a particular accent, rhythm, intonation style, pronounciation pattern, or choice of vocabulary.

2.2 Voice Activity Detection

Voice Activity Detection (VAD) is used in telecommunications, for example, in telephony to detect touch tones and the presence or absence of speech. Detection of speaker activity can be useful in responding to barge-in, for pointing to the end of an utterance in automated speech recognition, and for recognizing a word intended to trigger start of a service, application, event, or anything else that may be deemed useful.

2.3 MFCCs

Mel Frequency Cepstral Coefficients (MFCCs) are a feature widely used in automatic speech and speaker recognition. MFCCs are a representation of the short-term power spectrum of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency. Figure 1 shows procedure, how to calculate MFCCs.

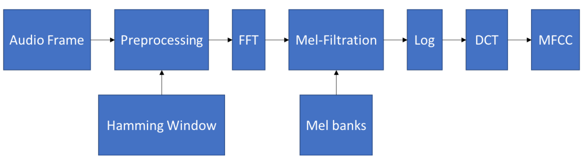

Figure 1. *Scheme of calculating MFCCs. In case of Kaldi recipe, Povey's window is used instead of Hamming window.*

Here, we can see a more detailed description of how to calculate MFCCs according to Figure 1:

1. Frame the signal into short frames.
2. For each frame calculate the periodogram estimate of the power spectrum.
3. Apply the mel filterbank to the power spectra, sum the energy in each filter.
4. Take the logarithm of all filterbank energies.
5. Take the DCT of the log filterbank energies.

Table 1. *The embedding DNN architecture. x-vectors are extracted at layer segment6, before the nonlinearity. 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 and finally the soft-max output layer. [3]*

Layer	Layer context	Total context
frame1	[t-2,t+2]	5
frame2	{t-2,t,t+2}	9
frame3	{t-3,t,t+3}	15
frame4	{t}	15
frame5	{t}	15
stats pooling	[0, T]	T
segment6	{0}	T
segment6	{0}	T
softmax	{0}	T

2.4 x-vector

Using deep neural networks (DNN) to capture speaker characteristics is currently a very active research area. The used system is a feed-forward DNN that computes speaker embeddings from variable-length acoustic segments and is based on [2, 3, 1].

The network 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 soft-max output layer, all layers with their respective contexts are shown in Table 1. The nonlinearities are rectified linear units (ReLUs). The network is trained to classify training speakers using a multi-class cross entropy objective function.

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. Therefore, any layer after the statistics pooling layer is a sensible place to extract the embedding from.

2.4.1 E-TDNN x-vector

The extended version of the TDNN described in 2.4, which is the default architecture in the public Kaldi recipes is described here. Table 2 summarizes the extended network (E-TDNN) architecture. The two main differences are a slightly wider temporal context of the TDNN (due to the addition of layer 7), and interleaving dense layers in between the convolutional layers (equivalent to the 1x1 convolutions used in computer vision architectures). This architecture has been found to greatly outperform the baseline TDNN in the SITW

Table 2. *Extended TDNN x-vector architecture.*

Layer	Layer Type	Layer context	Size
1	TDNN-ReLU	[t-2,t+2]	512
2	Dense-ReLU	t	512
3	TDNN-ReLU	{t-2, t, t+2}	512
4	Dense-ReLU	t	512
5	TDNN-ReLU	{t-3, t, t+3}	512
6	Dense-ReLU	t	512
7	TDNN-ReLU	{t-4, t, t+4}	512
8	Dense-ReLU	t	512
9	Dense-ReLU	t	512
10	Dense-ReLU	t	1500
11	Pooling (mean + stddev)	Full-seq	2x1500
12	Dense(Embedding)-ReLU		512
13	Dense-ReLU		512
14	Dense-SoftMax		512

and SRE16 benchmarks. The network outputs posterior probabilities for the training speakers and it was trained by minimizing a categorical cross-entropy. The x-vector is extracted from layer 12 prior to the ReLU non-linearity.

2.5 Backend

2.5.1 PLDA

To facilitate comparison of i-vectors and x-vectors in a verification trial, the distribution of i-vectors and x-vectors is modeled using a Probabilistic Linear Discriminant Analysis (PLDA) model [6, 7]. First, consider only a special form of PLDA, a *two-covariance model*, in which speaker and inter-session variability are modeled using across-class and within-class full covariance matrices Σ_{ac} and Σ_{wc}. The two-covariance model is a generative linear-Gaussian model, where latent vectors $\mathbf{y}$ representing speakers (or more generally classes) are assumed to be distributed according to prior distribution

$$p(\mathbf{y}) = \mathcal{N}(\mathbf{y}; \mu, \Sigma_{ac}). \tag{1}$$

For a given speaker represented by a vector $\hat{\mathbf{y}}$, the distribution of i-/x-vectors is assumed to be

$$p(\phi|\hat{\mathbf{y}}) = \mathcal{N}(\phi; \hat{\mathbf{y}}, \Sigma_{wc}). \tag{2}$$

The ML estimates of the model parameters, μ, Σ_{ac}, and Σ_{wc}, can be obtained using an EM algorithm as in [7].

In this paper, we will use gaussian and heavy-tailed [8] PLDA backend.

2.5.2 Diarization

Speaker diarization, the process of partitioning an audio stream with multiple people into homogeneous segments associated with each individual, is an important part of speech recognition systems. By solving the problem of *who spoke when*, speaker diarization has applications in many important scenarios, such as understanding medical conversations, video captioning and more. Example of diarization output is shown in Figure 2.

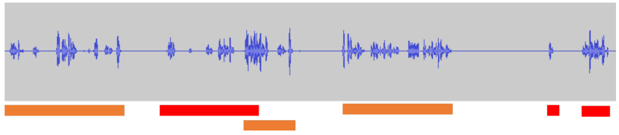

Figure 2. *Example output of diarization on single channel audio. Different colors in the bottom indicate different speakers.*

The used speaker diarization method is based on the Bayesian Hidden Markov Model described in [9], in which states represent speaker specific distributions and transitions between states represent speaker turns. The transitions probabilities are set to favor staying in the same speakers to avoid too frequent speaker turns. As in the ivector or JFA models, speaker distributions are modeled by GMMs with parameters constrained by eigenvoice priors to facilitate discrimination between speakers.

3. Experimental Setup

3.1 Data

All data we used either for training or testing purposes were data allowed by NIST for SRE18.

3.1.1 Training Data

Training data defines the amount and category of resources which are allowed to build speaker recognition system with. The training condition limits the system training to specific common data sets used for NIST SRE 2018, as shown in [1].

3.1.2 Evaluation Data

Since we are building robust speaker recognition system, we decided not to include some of the training corpora into the training set and use them for testing purposes instead. Specifically, we used all testing subsets from Speakers In The Wild (SITW) [10] and VoxCeleb1 [11]. Since SRE18 data are split into two main domains (narrowband and wideband), we decided to use only data, that match our testing conditions for Video Annotation for Speech Technology (VAST):

1. SITW core-core evaluation condition [10] (sitwEvalC-C)

[1] https://www.nist.gov/sites/default/files/documents/2018/08/17/sre18_eval_plan_2018-05-31_v6.pdf

2. SITW multi-core evaluation condition [10] (sitwEvalM-C)
3. VoxCeleb1 evaluation condition [11] (voxc1)
4. 2018 NIST SRE Development (dev) Set (LDC2018E46) VAST evaluation condition (sre18DevVAST)
5. 2018 NIST SRE Evaluation (eval) Set VAST evaluation condition (sre18EvalVAST)

3.2 Voice Activity Detection

VAD we used consists of two parts

- a neural network which produces per-frame scores and
- a postprocessing stage which builds the segments based on the scores.

For more information see [12].

We were only using generated VAD files, we were not running VAD system training.

3.3 x-vector

We used original features configuration of x-vector recipe [3] obtained from [2] - 23-dimensional filterbanks with a frame-length of 25ms, mean-normalized over a sliding window of up to 3 seconds. We slightly modified our voice activity detector from 3.2 and extended all speech frames by 15 frames to the left and also to the right, effectively extending the amount of speech that is passed into time-delay neural network, as shown in [13]. Also, we analyzed and applied some of the possible improvements for x-vector based architecture based on [13], such as larger number of augmentation (128 000 in original recipe vs. 256 000 in our recipe) and we also used larger number of epochs (3 in original recipe compared to 6 in our recipe) and this system will be used as our baseline x-vector system.

If not specified otherwise, we used 512-dimensional x-vector projected into 128-dimensional space using LDA. For scoring, we used gaussian PLDA backend.

We used the same data for x-vector training as in original recipe from [3].

3.4 Diarization

We used 19 MFCC+Energy coefficients (without any normalization) as features for diarization. We only ran the diarization on segments that contain speech according to our VAD. We used 1024-component, diagonal covariance GMM-UBM, and 400-dimensional i-vectors. The UBM and the total variability matrix were trained on the VoxCeleb1 and VoxCeleb2 datasets.

[2] https://david-ryan-snyder.github.io/2017/10/04/model_sre16_v2.html

A hierarchical agglomerative clustering (AHC) algorithm based on PLDA scores between i-vectors estimated on 200 ms segments was performed to initialize the assignment of frames to speakers for the VB algorithm.

We were only using generated diarization metafiles, we were not running diarization system training.

4. Experiments

In this chapter, we analyze the performance of our systems on wideband conditions.

We experimented with x-vector system and E-TDNN x-vector system, results for VAST (wideband) conditions are shown in Table 3. We also added BUT i-vector system using 16kHz data and MFCC as features from SITW evaluations [14]. We can see, that our x-vector system performs better than BUT system from final submission and E-TDNN based system highly outperforms even our modified x-vector architecture.

4.1 Domain Specific System

Here, we tried to adapt our system to target data during system training and therefore use only wideband data for system training, since development corpus for SRE18 VAST condition is very small and not statistically reliable. For training we used VoxCeleb1 [11] and VoxCeleb2 [15] training sets, we trained extractor (x-vector NN) and also PLDA model on the same set.

We used the following modifications compared to original recipe [3] for all our experiments:

- 9 epochs instead of 3 in the original recipe
- total 512 000 augmentations instead of 128 000 in the original recipe
- concatenate all utterances from a single session with one second of silence between every utterance.

Results for domain-specific systems are shown in Table 4. When we compare these results to results in Table 3, we can see that using domain-specific data is crucial for system's performance and even with our best E-TDNN system trained on telephone data with EER 5.90% on sitwEvalC-C we are not competitive with baseline x-vector system trained on wideband data with EER 4.89%.

In our experiments, we slightly changed the topology of TDNN to accept larger context, these modifications are shown in Table 5 and are marked with suffix *LC (large context)*. We can conclude, that extending the context of TDNN improved results in terms of EER and also for another operating point. Also, we can see a very significant gain in using 16k sample rate over

Table 3. *Baseline results on VAST-similar datasets without using diarization.*

System	sitwEvalC-C		voxc1	
	EER[%]	$\mathrm{DCF}_{0.01}^{\min}$	EER[%]	$\mathrm{DCF}_{0.01}^{\min}$
BUT i-vector [14]	9.34	0.713		
x-vector	7.16	0.559	9.00	0.676
E-TDNN	5.90	0.519	7.74	0.599

8k sample rate - for competitive systems x-vector LC with 8k sample rate and 16k sample rate respectively, we can see almost 30% relative improvement in terms of EER. As described in 2.5.1, we also used HT PLDA backend for E-TDNN system (E-TDNN HT-PLDA) and using this setup, we were able to obtain best results for the sitwEvalC-C condition with 2.13% EER and $\mathrm{DCF}_{0.01}^{\min}$ 0.221.

Detection Error Tradeoff (DET) curve for corresponding systems on sitwEvalC-C condition is in Figure 3. We can conclude, that E-TDNN system outperforms standard x-vector architecture with extended context but both systems are competitive for false acceptance ratio under 1%.

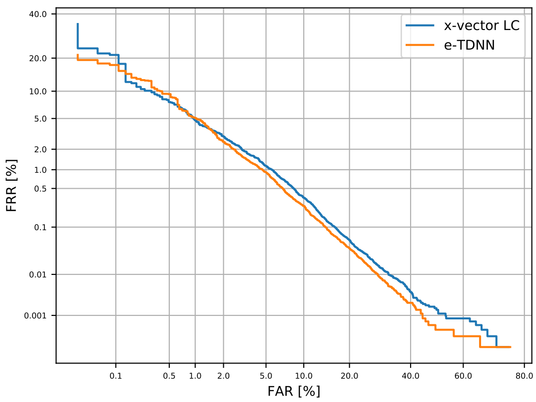

Figure 3. *Detection error tradeoff curve for systems trained on 16k Hz VoxCeleb1 and VoxCeleb2 data for sitwEvalC-C condition.*

4.2 Diarization in the Loop

In this section we analyze the performance of our system on testing conditions which necessarily does not contain single speakers at enroll or test sides, therefore it should be sensible to run automatic diarization systems before performing speaker verification. We analyze the performance of our best systems with and without diarization, results are shown in Table 6. For all our experiments we used the diarization system described in 3.4.

DET curve for sre18EvalVAST condition is shown in Figure 4. DET curves show us, that there is a very small difference between the x-vector LC system and E-TDNN, evaluation dataset is still very small and results may be noisy. We can conclude, that diarization helps for all our systems on sitwEvalM-C condition by 20% in terms of EER and also by 20% for $\mathrm{DCF}_{0.01}^{\min}$. On sre18EvalVAST condition however, there is almost no gain in performance when using diarization.

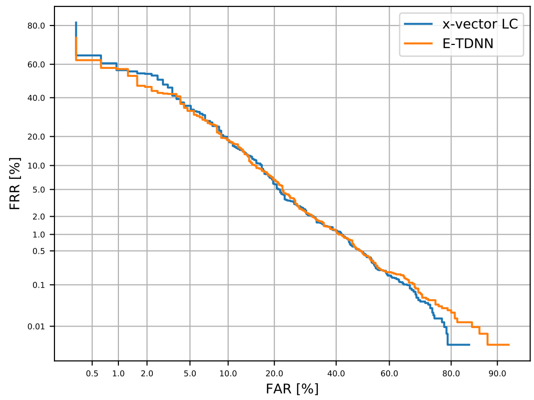

Figure 4. *Detection error tradeoff curve for systems trained on VoxCeleb1 and VoxCeleb2 data for sre18EvalVAST condition using diarization marks and enrollment annotations.*

5. Conclusions

In this experimental work, we analyzed the state-of-the-art speaker verification pipeline using x-vector based speaker embeddings. We show, that using in-domain wideband data for training, in this case, VoxCeleb1 and VoxCeleb2, we were able to outperform systems trained on 8kHz data. VoxCeleb1 and VoxCeleb2 datasets are also very large, containing over 1 million utterances from thousands of speakers and allow us to use state-of-the-art deep learning methods.

We also experimented with improving our scoring backend and we used Heavy Tailed PLDA for scoring, yielding 2.13% EER on the sitwEvalC-C dataset, using an out-of-the-box system, without any adaptation to SITW dataset. Comparing our results on voxc1 test dataset to ResNet architecture from [15], in terms of equal error rate using E-TDNN with HT-PLDA backend we obtained 2.73% EER compared to their 3.95%.

Also, our best wideband system produced during

Table 4. *Results for domain-specific systems on VAST-similar datasets without using diarization.*

System	Sample Rate	sitwEvalC-C		voxc1	
		EER[%]	$\mathrm{DCF}^{\mathrm{min}}_{0.01}$	EER[%]	$\mathrm{DCF}^{\mathrm{min}}_{0.01}$
x-vector	8k	4.89	0.448	6.61	0.634
x-vector LC	8k	3.85	0.392	5.22	0.56
x-vector LC	16k	2.74	0.268	2.99	0.33
E-TDNN	16k	2.60	0.242	2.77	0.286
E-TDNN HT-PLDA	16k	**2.13**	**0.221**	2.73	0.304

Table 5. *Configuration of TDNN for x-vector extraction using larger context. Bold values are our modifications of the original [3] architecture. X-vectors are extracted at layer segment6 before the nonlinearity.*

Layer	Layer context	Total context
frame1	[t-2,t+2]	5
frame2	{**t-4**, t-2,t,t+2,**t+4**}	13
frame3	{**t-6**,t-3,t,t+3,**t+6**}	19
frame4	{t}	19
frame5	{t}	19
stats pooling	[0, T]	T
segment6	{0}	T
segment7	{0}	T
softmax	{0}	T

NIST SRE 2018 evaluations was used as one of the submission systems and was very competitive considering all submissions of other teams. Using diarization in speaker verification, however, still looks like a problematic area with very high error rates and should be also included as an active area of speech technology research.

Our future work will be focused on experimenting more with E-TDNN architecture, such as extending the context of time-delay layers and stacking more of these layers into a network.

Acknowledgements

I would like to thank my supervisor Ing. Pavel Matějka PhD. for his extensive support. I would like to also thank MSc. Anna Silnova for her help with HT-PLDA implementation and also to my colleagues Mgr. Josef Slavíček and Ing. Michal Klčo.

References

[1] David Snyder, Pegah Ghahremani, Daniel Povey, Daniel Garcia-Romero, Yishay Carmiel, and Sanjeev Khudanpur. Deep neural network-based speaker embeddings for end-to-end speaker verification. In *Spoken Language Technology Workshop (SLT), 2016 IEEE*, pages 165–170. IEEE, 2016.

[2] David Snyder, Daniel Garcia-Romero, Daniel Povey, and Sanjeev Khudanpur. Deep neural network embeddings for text-independent speaker verification. In *Proc. Interspeech*, pages 999–1003, 2017.

[3] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur. X-vectors: Robust dnn embeddings for speaker recognition. *Submitted to ICASSP*, 2018.

[4] Najim Dehak, Patrick J Kenny, Réda Dehak, Pierre Dumouchel, and Pierre Ouellet. Front-end factor analysis for speaker verification. *IEEE Transactions on Audio, Speech, and Language Processing*, 19(4):788–798, 2011.

[5] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukáš Burget, Ondřej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, Jan Silovský, Georg Stemmer, and Karel Veselý. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society, December 2011. IEEE Catalog No.: CFP11SRW-USB.

[6] S. J. D. Prince and J. H. Elder. Probabilistic linear discriminant analysis for inferences about identity. In *11th International Conference on Computer Vision*, pages 1–8, 2007.

[7] Patrick Kenny. Bayesian speaker verification with heavy-tailed priors. In *Odyssey*, page 14, 2010.

[8] Anna Silnova, Niko Brummer, Daniel Garcia-Romero, David Snyder, and Lukáš Burget. Fast variational bayes for heavy-tailed plda applied to i-vectors and x-vectors. *arXiv preprint arXiv:1803.09153*, 2018.

[9] Mireia Diez, Lukáš Burget, and Pavel Matějka. Speaker diarization based on bayesian hmm with

Table 6. *Results for domain specific systems on VAST-similar datasets.*

System	Diarization	sitwEvalM-C		sre18EvalVAST	
		EER[%]	$DCF_{0.01}^{min}$	EER[%]	$DCF_{0.01}^{min}$
x-vector LC	no	5.20	0.363	13.33	0.746
E-TDNN	no	5.09	0.338	13.33	0.758
x-vector LC	yes	4.14	0.292	13.59	0.713
E-TDNN	yes	4.02	0.269	12.35	0.738

eigenvoice priors. In *Odyssey 2018, The Speaker and Language Recognition Workshop*, 2018.

[10] Mitchell McLaren, Luciana Ferrer, Diego Castan, and Aaron Lawson. The 2016 speakers in the wild speaker recognition evaluation. In *INTERSPEECH*, pages 823–827, 2016.

[11] Arsha Nagrani, Joon Son Chung, and Andrew Zisserman. Voxceleb: a large-scale speaker identification dataset. *arXiv preprint arXiv:1706.08612*, 2017.

[12] Pavel Matějka, Oldřich Plchot, Ondřej Novotný, Sandro Cumani, Alicia Lozano-Diez, Josef Slavicek, Mireia Diez, František Grézl, Ondřej Glembek, Kamsali Veera Mounika, et al. But-pt system description for nist lre 2017.

[13] Mitchell McLaren, Diego Castan, Mahesh Kumar Nandwana, Luciana Ferrer, and Emre Yılmaz. How to train your speaker embeddings extractor. In *Odyssey: The Speaker and Language Recognition Workshop, Les Sables d'Olonne*, 2018.

[14] Ondřej Novotný, Pavel Matějka, Oldřich Plchot, Ondřej Glembek, Lukáš Burget, and Jan Cernocký. Analysis of speaker recognition systems in realistic scenarios of the sitw 2016 challenge. In *Interspeech*, pages 828–832, 2016.

[15] Joon Son Chung, Arsha Nagrani, and Andrew Zisserman. Voxceleb2: Deep speaker recognition. *arXiv preprint arXiv:1806.05622*, 2018.