Odyssey 2018 The Speaker and Language Recognition Workshop
26-29 June 2018, Les Sables d'Olonne, France

On deep speaker embeddings for text-independent speaker recognition

Sergey Novoselov[1,2], *Andrey Shulipa*[1], *Ivan Kremnev*[2], *Alexandr Kozlov*[2], *Vadim Schemelinin*[1]

[1]ITMO University, St.Petersburg, Russia
[2]STC-innovations Ltd., St.Petersburg, Russia

{novoselov,shulipa,kremnev,kozlov-a,shchemelinin}@speechpro.com

Abstract

We investigate deep neural network performance in the text-independent speaker recognition task. We demonstrate that using angular softmax activation at the last classification layer of a classification neural network instead of a simple softmax activation allows to train a more generalized discriminative speaker embedding extractor. Cosine similarity is an effective metric for speaker verification in this embedding space. We also address the problem of choosing an architecture for the extractor. We found that deep networks with residual frame level connections outperform wide but relatively shallow architectures. This paper also proposes several improvements for previous DNN-based extractor systems to increase the speaker recognition accuracy. We show that the discriminatively trained similarity metric learning approach outperforms the standard LDA-PLDA method as an embedding backend. The results obtained on Speakers in the Wild and NIST SRE 2016 evaluation sets demonstrate robustness of the proposed systems when dealing with close to real-life conditions.

1. Introduction

Text-independent speaker recognition continues to be a challenging task for modern voice biometrics systems. Complex speaker voice information must be captured from highly variate data with no evident speaker patterns. Candidate solutions must generalize well in order to be applicable to new possible deployment conditions. This work investigates prominent techniques from speaker recognition field combined with face recognition and general deep learning science to bring new thoughts on how speaker recognition systems can be developed.

I-vector-based systems are well known to be state-of-the-art solutions to the text-independent speaker verification problem [1, 2, 3]. The i-vector framework has inspired deep learning system design in this field. Particularly, in studies [2, 4] they use an ASR deep neural network (ASR DNN) to divide acoustic space into senone classes, and the classic total variability (TV) model is applied to discriminate between speakers in that space [1]. In such phonetic discriminative DNN-based systems two major techniques can be distinguished. Firstly, one can use DNN posteriors to calculate Baum-Welch statistics, and secondly, bottleneck features extracted with a network can be used in pair with speaker specific features (MFCC) for a full TV-UBM system training.

Deep learning is a powerful tool for data analysis with complex data distribution [5, 6, 7, 8], and many researches consider training deep non-linear extractors as a solution to the direct speaker discrimination task. There are several solid studies on advantageous usage of deep end-to-end solutions for discriminating speakers directly in a text-dependent task [9, 10, 11]. Paper [12] describes a deep network that extracts small speaker footprints which are used to discriminate between speakers.

Paper [13] presents a rather powerful implementation of a DNN speaker embedding extractor based on the speaker discriminative approach in the text-independent setting. One of the key features of the system proposed by its authors is the time-delay neural network architecture of the extractor [14] with a statistics pooling layer designed to accumulate speaker information from the whole speech segment into a single vector, which they call an x-vector. X-vectors expose similarities to i-vectors from a total variability space, and so can be effectively used with the standard Linear Discriminant Analisys (LDA) followed by Probabilistic Linear Discriminant Analisys (PLDA) backend for solving the speaker recognition task. The authors show that a discriminatively trained x-vector extractor surpasses the standard i-vector extractor in terms of speaker recognition error on the challenging NIST 2016 test.

This study also speculates on the discriminative approach for learning a deep speaker embedding extractor in the text-independent scenario. This problem is usually solved by training a deep neural network speaker classifier on a closed speaker set and using its bottleneck features as speaker embeddings after. These embeddings are believed to contain informative speaker factors and respond moderately to any other speech variation if trained effectively.

A feature extractor trained this way must generalize well in order to be applicable to an open-set speaker verification. It is challenging to achieve this property only using standard first order optimization algorithms and standard cross-entropy loss for softmax outputs of the network. The main reason for that is quick overfitting of such a system to corpus-specific speech features like the language, the audio channel type and quality and so on. Unfortunately, common regularization techniques like weight decay and dropout do not help much to resolve these issues.

However, according to our observations, more complex optimizators can be a solution to the problem, e.g. natural gradient, as was done for the x-vector extraction procedure. They allow the algorithm to converge to a lower optimum while being trained for multiple epochs.

Another option for building a more generalized extractor is a better choice of the loss function objective for the training stage and appropriate design of the architecture of a neural network.

Thus in this study we ponder on the right architecture for the neural network and a proper choice of the loss function. We take some ideas from the field of face recognition and general deep learning to our advantage and train a speaker embedding

10.21437/Odyssey.2018-53

extractor that is compatible with the x-vector system but is computationally lighter due to the lower number of parameters.

In addition, we study performance of the x-vector-based system with a learnable Cosine Similarity Metric (CSML) backend [15], which outperforms the standard LDA-PLDA backend for x-vectors.

The rest of the paper is organized as follows. Section 2 describes previous systems and techniques that we use as baseline or as a component of our systems. Section 3 describes highlights of each deep extractor we train and study along with their configuration presented in Tables 1, 2. Section 4 describes details of the whole models, including frontend feature configuration and backend embedding classifiers. Sections 5 and 6 cover experimental setups and results, and Section 7 speculates on strong points of each system and directions for further development of text-independent speaker recognition systems.

2. Related work

This section describes state-of-the-art speaker recognition systems, including i-vector baselines and a DNN-based speaker embedding extractor.

2.1. Baseline i-vectors

Most of the text-independent speaker recognition systems are based on the i-vector extraction framework. Typically, i-vector computation process can be decomposed into three stages: collection of sufficient statistics, extraction of i-vectors and a probabilistic linear discriminant analysis (PLDA) backend [2, 1, 4]. Sufficient statistics are collected by using a sequence of feature vectors, e.g. melfrequency cepstral coefficients (MFCC), and are usually represented by Baum-Welch statistics obtained with respect to a GMM, which is called a universal background model (UBM). These statistics are converted into a single low-dimensional feature vector — an i-vector — that contains substantial information about the speaker and all other types of speech variability. After i-vectors are extracted, a PLDA model is used to produce verification scores by comparing i-vectors extracted from different speech segments.

An alternative i-vector framework is based on deep neural networks. DNN/i-vector frameworks provides the best speaker recognition performance in "clean" speech conditions [2, 16]. In the DNN-based i-vector framework a deep neural network substitutes a UBM in calculation of Baum-Welch statistics, followed by total variability factor analysis. Alternatively, DNN can be used for extracting bottleneck (BN) features, which are used together with MFCC in the standard UBM/i-vector framework. This also gives impressive speaker recognition performance [17].

A drawback of the latter systems is their low robustness when applied in real acoustic settings. This is confirmed with the NIST 2016 SRE results [18]. In such setting standard acoustic i-vector is preferred.

In our experiments we use only DNN posterior-based i-vector extraction procedure for baseline.

2.2. DNN speaker embeddings

Impressive results can be obtained by using solely a deep neural network classifier as an extractor of speaker embeddings. Trained on basic speech features like log mel power signal spectrum to discriminate between speaker classes, such network is a convenient tool for extracting high-level speaker features from top levels of the network.

One example of such an extractor is the x-vector system [13]. The systems uses MFCC features of the speech signal of a fixed as input features for a TDNN-type neural network. Each time-delay layer of the network gradually develops speaker-informative features on the feature map while also expanding context of a frame which is crucial for global speaker capturing. Time-delay layers are followed by a stats pooling layer, which folds features along the time axis by calculating mean and standard deviation statistics of those features. As a result, we get a highly representative feature vector containing global speaker information. To get speaker embedding right, this vector is passed through several fully-connected layers of a classifier network.

It is shown that x-vector-based systems outperform i-vector-based systems when applied to in-the-wild conditions [13, 19]. However, using softmax cross entropy loss function for extractor training does not allow to use standard metrics, such as cosine metric, for embedding scoring. In this case, an LDA-PLDA backend gives good results.

The detailed configuration for the x-vector extractor is presented in [13].

2.3. Angular softmax loss with margin

There are known many heuristics for effective discriminative learning of a deep extractor from the face recognition field. Most widely used regularized loss functions are centerloss with softmax loss [20], contrastive loss [21] and triplet loss [22, 23]. We believe that a robust speaker recognition system can be designed to extract embeddings onto a hypersphere manifold with a cosine similarity metric suitable for their comparison. An example of this representation is the commonly used i-vector normalization technique. Work [8] demonstrates an elegant way to obtain well regularized loss function by forcing learned features to be discriminative on a hypersphere manifold. This idea directly leads to a rigorous formulation of the angular margin softmax loss:

$$L_{\text{ang}} = \frac{1}{N}\sum_i -\log\Big(\frac{e^{\|x_i\|\cos(m\theta_{i,y_i})}}{e^{\|x_i\|\cos(m\theta_{i,y_i})} + \sum_{i\neq y_i} e^{\|x_i\|\cos(m\theta_{i,y_i})}}\Big)$$

where N is the number of training samples $\{x_i\}_{i=1}^N$ and their labels $\{y_i\}_{i=1}^N$, θ_{i,y_i} is the angle between x_i and the corresponding column y_i of the fully connected classification layer weights $\mathbf{W}$, and m is an integer that controls the size of an angular margin between classes.

We use Asoftmax as an effective discriminative objective for training our deep embedding extractor.

3. Analisys of deep learning systems

Architecture choice is crucial for a neural network to act as a powerful speaker embedding extractor since one must address capturing speaker information from both local and global patterns in speech signals. There also several requirements for a networks to be applicable in real tasks: its computational lightness and high generalization. Therefore we consider two alternative architectures.

3.1. Max pooling embeddings

We introduce a modification of the original TDNN-based x-vector extractor [13] with intermediate max pooling layers, which we call a max pooling embedding extractor. Max pooling layers are very common within convolutional neural networks and are a source of spatial invariance and dimensionality reduction of data for algorithms in addition to necessary non-linearity. Including intermediate max pooling layers effectively turns a TDNN architecture to a more CNN-like. It reduces amount of network parameters and speeds up computations.

Time-delay layers use PReLU [24] instead of the conventional ReLU activation function. For the segment level of the network, which comes after the stats pooling layer, we also make use of an alternative activation function. Here we use Max-Feature-Map (MFM) activation [25] in place of ReLU. In contrast to commonly used ReLU function that suppresses neurons by a zero threshold, MFM suppresses neurons by mutually competitive relationships. By doing so the MFM activation acts as an embedded feature selector.

Our x-vector modification architecture is decribed in Table 1. Max pooling is made with windows are 2×2 with stride 2. N_{spk} is the number of speakers in training set. Layer context can be thought as of width of a convolution filter along time dimension if expressed in terms of CNNs.

Table 1: Max pooling embedding extractor configuration. Frame layers correspond to the TDNN architecture part of the network, while segment layers to the fully-connected one. Stats pooling layer is the intermediate time-folding layer.

Layer	**Layer context**	**Total context**	**In $\times$ out**
frame1	7	7	161 $\times$ 256
maxpool1	2	8	256 $\times$ 128
frame2	5	18	640 $\times$ 256
maxpool2	2	20	256 $\times$ 128
frame3	3	28	384 $\times$ 256
maxpool3	2	32	256 $\times$ 128
frame4	1	32	256 $\times$ 2048
maxpool4	2	32	2048 $\times$ 1024
stats pooling	all	all	1024 $\times$ 2048
segment6 MFM	all	all	2048 $\times$ 1024
segment7 MFM	all	all	1024 $\times$ 512
A-softmax	all	all	512 $\times$ N_{spk}

It can be seen that frame-level layers are responsible for capturing time-local speaker features, while stats pooling collects global information.

3.2. Deep residual embeddings

The necessity of wide context capturing seems essential when trying to collect informative speaker features and separate them from other signal variations. There are two ways of context expanding in TDNN architecture: either by widening it at each frame-level layer, or by deepening the network to accumulate richer context with higher level of feature abstraction.

Our second alternative architecture is a deep extractor consisting of time-delay layers with shallow frame-level contexts. A practical way of training a rather deep network is additional of residual connections, which were initially introduced in paper [26] for deep image representation learning. Hence we introduce a residual TDNN block, illustrated by Figure 1. A technical detail is that we need to ensure that the dimensions match when adding residual part to output in a residual block. For this purpose zero padding was applied.

Table 2: Deep residual embedding extractor configuration. N_{spk} is the number of speaker classes, which determines the number of neurons at the output layer.

Layer	**Layer context**	**Total context**	**In $\times$ out**
frame1	3	3	69 $\times$ 128
maxpool1	2	4	128 $\times$ 64
frame2: resTDNN block 1	3	8	64 $\times$ 64
...			
frame $M+1$: resTDNN block M	3	$8M$	64 $\times$ 64
frame $M+2$	1	$8M$	64 $\times$ 2048
maxpool $M+2$	2	$16M$	2048 $\times$ 1024
stats pooling	all	all	1024 $\times$ 2048
segment6 MFM	all	all	2048 $\times$ 1024
segment7 MFM	all	all	1024 $\times$ 512
A-softmax	all	all	512 $\times$ N_{spk}

A big advantage of using residual connections is the ability to pass primitive features to top layers. The network itself adapts to the level of abstraction needed at each layer by leveraging weights, and so the effective width of the context for each level is also adapted from the speech setting.

Our deep residual TDNN extractor contains M residual blocks, a stats pooling layer for feature aggregating through time and a fully-connected classifier. The network architecture is described precisely in Table 2.

3.3. Backends

In our experiments we measured quality of speaker embeddings with a backend and without one. In the last case simple cosine similarity metric can be applied for verification directly in the embedding space. Moreover, the discriminative metric learning approach could be viewed as a scalable alternative to simple cosine metric or LDA-PLDA backend for deep speaker embeddings.

In this work we study how Cosine Similarity Metric Learning can improve embedding verification only for x-vectors based system. In CSML, a linear transformation $\mathbf{A}$ must be learned to compute cosine similarities (CS) on a pair $(\mathbf{x_1}, \mathbf{x_2})$ as follows:

$$\mathcal{S}(\mathbf{x_1}, \mathbf{x_2}, \mathbf{A}) = \frac{(\mathbf{A}\mathbf{x_1})^T(\mathbf{A}\mathbf{x_2})}{\|\mathbf{A}\mathbf{x_1}\|\|\mathbf{A}\mathbf{x_2}\|} \tag{1}$$

where the transformation matrix $\mathbf{A}$ is upper triangular. Under this constraint $\mathbf{A}^T\mathbf{A}$ is positive-definite. Unlike [15] we set the triplet loss objective function for training $\mathbf{A}$:

$$\mathcal{L}(\mathbf{A}) = \sum_{a,p,n \in T} \log(1 + \exp(-d_{a,p,n})) \tag{2}$$

where $d_{a,n,p} = s_{a,p} - s_{a,n}$ is the difference between similarity scores $s_{a,p}$ and $s_{a,n}$. T is a collection of training triplets which is formed from a training dataset. A triplet (a, p, n) contains an anchor sample a as well as a positive $p \neq a$ and a negative n example of the anchor's identity. We included in T all positive

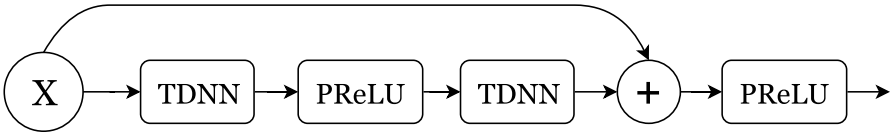

Figure 1: TDNN Residual block

examples and only 1500 hardest negative examples for selected anchors.

4. Implementation details

This section describes details of the studied speaker recognition systems.

4.1. DNN posterior-based i-vectors

We extracted DNN posterior-based i-vectors as described in 2.1 for a baseline. A DNN was trained on the Switchboard corpus using the Kaldi speech recognition toolkit [27]. Outputs of the DNN correspond to the set of 2700 speech triphone states. These 2700 speech-related outputs were used for statistics calculation. 20 MFCCs (including log energy) were calculated using 23 filter banks with their first and second derivatives. Mean and variance normalization was subsequently applied.

This system is called DNN/i-vector.

4.2. X-vectors

We used the same x-vector based system configuration as described in [13] and available in Kaldi [27]. Our system takes 23 dimensional MFCCs as input instead of raw filterbanks. MFCCs are computed with a frame-length of 25ms, mean-normalized over a 3-second sliding window. Energy-based voice activity detector (VAD) filters out non-speech frames.

The speaker embeddings are extracted from the second to last layer of the classifier network. We applied Cosine Similarity Metric Learning approach (see 3.3) instead of a LDA-PLDA backend as an alternative embedding classifier.

We refer to this system as X-vectorNet.

4.3. Speaker Max-Pooling Net

Our first development is a max pooling TDNN-based speaker embedding extractor, which is described in Section 3.1. Max-Pooling TDNN uses the same front-end features as x-vector based system in section 4.2. The network is trained on short segments of speech (3-10 sec), which are randomly sampled from the training data.

After classificator training, the last fully-connected layer with its angular softmax activation is removed from the network in order to obtain an extractor of high-level representations for speaker specific information. Simple cosine metric as well as LDA-PLDA approach can be used as a backend to the Max-pooling TDNN embeddings.

The model was implemented in PyTorch [28] and trained using a single GeForce GTX 1080 GPU.

We refer to this system as SpeakerMaxPoolNet.

4.4. Speaker Residual Net

Our deepest embedding extractor architecture is represented by a deep neural network with TDNN residual blocks, as described in Section 3.2. It takes the same MFCC input features as X-vectorNet and SpeakerMaxPoolNet. The network is trained on short random segments of speech (3-10 sec), which are randomly sampled from the training data.

The embeddings are also extracted from the penultimate fully-connected layer, as it is done in 4.3. Again, cosine metric as well as LDA-PLDA approach can be used as a backend to the Speaker Residual Net speaker embeddings.

The model was also implemented in PyTorch [28] and trained using a single GeForce GTX 1080 GPU.

We refer to this system as SpeakerResNet.

5. Experimental setup

5.1. Training data

We have prepared multiple training settings during our series of experiments. For preliminary studies, we used NIST 1998-2008 datasets for training with no data augmentation.

In our main experimental setup, telephone speech is collected as training data. It includes Switchboard2 Phases 1, 2, and 3, Switchboard Cellular and data from NIST SREs from 2004 through 2010. In addition, we use data augmentation as it was done in [19] to increase amount and diversity of the training data. In total, there are about 55,000 recordings from 5,277 speakers in this training part, a major part of which is English speech. We refer to this data as *English* data.

In other experiments we also used Russian speech subcorpus named RusTelecom to extend the training set. RusTelecom is a proprietary Russian speech corpus of telephone speech, collected by call-centers in Russia. The train part of the RusTelecom database consists of approximately 70000 sessions from 11087 speakers. We refer to this data as *Russian* data.

5.2. Evaluation

For preliminary studies, we used NIST 2010 evaluation dataset for testing under the det5 protocol.

Our main experimental setup includes evaluation on the Speaker-in-the-Wild [29] (SITW) and NIST SRE 2016 [18] datasets. In the case of NIST SRE 2016 we used the unequalized protocol.

We report results in terms of equal error rate (EER) and the minimum detection cost function (DCF) with $P_{\text{Target}} = 10^{-2}$ and $P_{\text{Target}} = 10^{-3}$.

6. Experimental results

6.1. Preliminary investigation

6.1.1. Angular Softmax vs regular Softmax

We compare effectiveness of the regular softmax and the margin-based A-softmax cross-entropy losses in training a deep speaker embedding extractor. Figure 2 shows NIST 2010 det5 protocol EER to the number of training iterations. We experiment with the two proposed architecture solutions, namely

Table 3: NIST 2010 det5 protocol evaluation results.

System	Backend	EER, %	DCF10^{-3}
DNN/i-vector	LDA-PLDA	1.67	0.3415
SpeakerMaxPoolNet7	cos	3.65	0.5866
SpeakerMaxPoolNet7	LDA-PLDA	3.71	0.5836
SpeakerResNet24	cos	3.01	0.498
SpeakerResNet24	LDA-PLDA	3.14	0.5131
SpeakerResNet44	cos	2.72	0.4967
SpeakerResNet44	LDA-PLDA	2.76	0.5256

SpeakerMaxPoolNet and SpeakerResNet.

These results demonstrate lack of generalization when using regular softmax for training. In contrast, margin-based A-softmax objective leads to comparatively good speaker generalization in the obtained discriminative speaker embedding space. It should be noted that simple cosine scoring was used for calculating system performance. Application of more complex backends such as LDA-PLDA slightly improves the results for embeddings trained with the regular softmax.

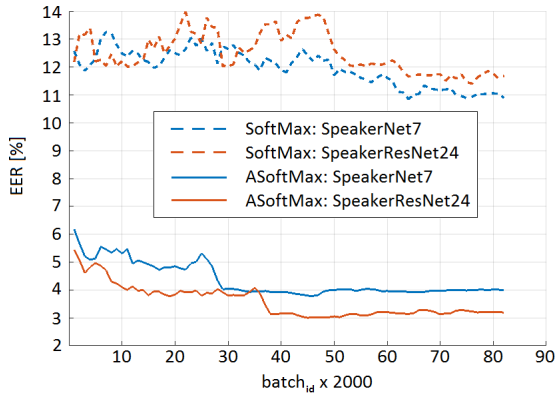

Figure 2: Comparison of speaker recognition performance on NIST 2010 det5 protocol for softmax and a-softmax classifiers used during traning. The numbers in the labels indicate the total amount of layers in the extractor.

6.1.2. *"Clean" condition experiments*

The preliminary experiment results in the "clean" telephone speech conditions are presented in Table 3. The NIST2010 evaluation protocol was used for testing. Again, we focus on SpeakerMaxPoolNet and SpeakerResNet architectures. A-softmax margin cross-entropy loss was applied to train the networks.

Figures 3, 4 illustrate EER and minDCF^{-10} evolution during DNN training. The results in Table 3 and Figures 3, 4 show that deep networks with residual frame-level connections are superior to wide but relatively shallow architectures. Unfortunately, these systems were not able to surpass DNN/i-vector baseline system in terms of quality in "clean" conditions.

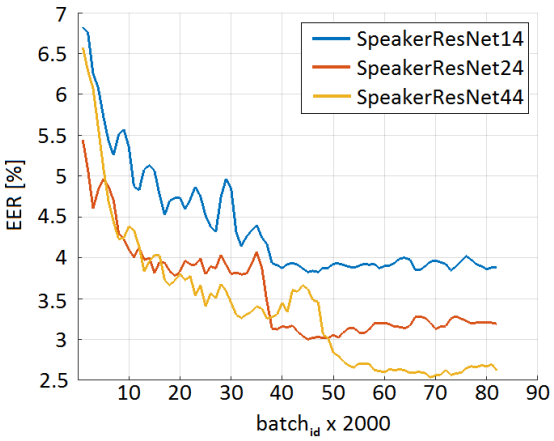

Figure 3: EER evolution on NIST 2010 det5 protocol for different architectures. The numbers in the labels indicate the total amount of layers in the extractor.

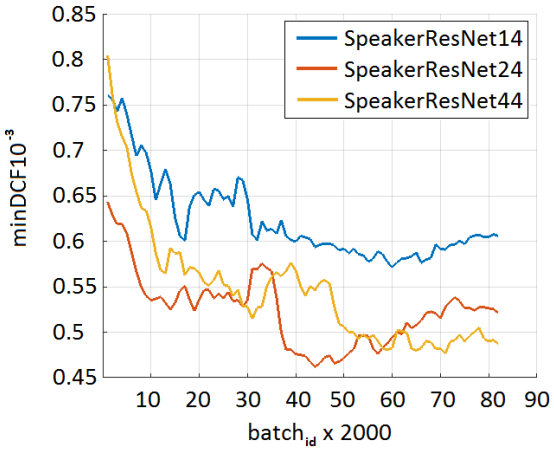

Figure 4: MinDCF10^{-3} evolution on NIST 2010 det5 protocol for different architectures. The numbers in the labels indicate the total amount of layers in the extractor.

6.2. Main experiments

In the main experiments we tested speaker recognition systems against "in the wild" conditions. To do this we trained DNN extractors on large augmented corpora. We tested two alternative extractors: SpeakerMaxPoolNet with 7 layers and SpeakerResNet with 24 layers.

According to the results of [13, 19], x-vector based systems with LDA-PLDA model backend are superior to standard i-vector based systems. Thus we used x-vector LDA-PLDA solution as a baseline in this case. Note that we did not used any embedding adaptation methods apart from centering on in-domain development set.

Tables 4, 5 present the results for systems trained only on *English* corpora. One can observe that SpeakerMaxPoolNet7 and SpeakerResNet24 perform well, despite simple cosine similarity scoring was applied. Moreover, the systems which were trained with margin angular softmax loss function outperform the x-vector baseline. We observe that the x-vector based LDA-PLDA system needs in-domain centering more than the other studied systems (see Table 5). Also, using Cosine Similarity

System	Backend	NIST2016			SITW		
		EER, %	DCF10^{-2}	DCF10^{-3}	EER, %	DCF10^{-2}	DCF10^{-3}
X-vectorNet	LDA-PLDA	16.88	1.00	1.00	7.82	0.6136	0.7753
X-vectorNet	CSML	**13.26**	**0.8686**	0.9972	8.58	0.5916	0.7487
SpeakerMaxPoolNet7	cos	14.57	0.9340	0.9930	7.72	0.5573	0.7320
SpeakerResNet24	cos	14.18	0.9257	**0.9881**	**6.72**	**0.5200**	**0.7156**

Table 4: Results using *English* corpora for training. No adaptation implemented.

System	Backend	NIST2016			SITW		
		EER, %	DCF10^{-2}	DCF10^{-3}	EER, %	DCF10^{-2}	DCF10^{-3}
X-vectorNet	LDA-PLDA	11.89	0.85	1.00	6.57	0.6031	0.7870
X-vectorNet	CSML	**10.97**	**0.7025**	0.9294	8.06	0.5947	0.7490
SpeakerMaxPoolNet7	cos	11.50	0.7545	0.9241	6.78	0.5413	0.7152
SpeakerResNet24	cos	11.13	0.7332	**0.8963**	**6.18**	**0.5079**	**0.7066**

Table 5: Results using *English* corpora for training. Centering on in-domain devset implemented.

System	Backend	NIST2016			SITW		
		EER, %	DCF10^{-2}	DCF10^{-3}	EER, %	DCF10^{-2}	DCF10^{-3}
X-vectorNet	LDA-PLDA	15.03	0.9974	1.00	11.62	0.7722	0.8971
X-vectorNet	CSML	**12.87**	**0.8602**	0.9894	9.62	0.6318	0.7861
SpeakerMaxPoolNet7	cos	13.09	0.8811	0.9879	7.35	0.5768	0.7585
SpeakerResNet24	cos	13.94	0.8937	**0.9869**	**7.08**	**0.5351**	**0.7025**

Table 6: Results using *English* and *Russian* datasets for training. No adaptation implemented.

System	Backend	NIST2016			SITW		
		EER, %	DCF10^{-2}	DCF10^{-3}	EER, %	DCF10^{-2}	DCF10^{-3}
X-vectorNet	LDA-PLDA	12.30	0.8732	1.00	11.73	0.7802	0.8984
X-vectorNet	CSML	**10.45**	**0.6914**	0.9235	8.70	0.6216	0.7998
SpeakerMaxPoolNet7	cos	11.26	0.7311	0.9241	6.40	0.5397	0.7236
SpeakerResNet24	cos	11.16	0.7128	**0.9024**	**5.90**	**0.5125**	**0.6987**

Table 7: Results using *English* and *Russian* datasets for training. Centering on in-domain devset implemented.

Metric Learning backend leads to significant performance improvement in comparison to LDA-PLDA.

Tables 6, 7 show system performance when an extended dataset containing *English* and *Russian* corpora is used for training. A notable average performance improvement for the SpeakerMaxPoolNet7 and SpeakerResNet24 based systems on both SITW and NIST 2016 evaluation is seen. In contrast, the x-vector based systems experience some degradation on SITW protocol.

7. Discussion

In our investigations we explored different strategies for discriminative speaker extractor training on the closed set task. We found that first-order optimization for regular softmax objective such as stochastic gradient descend always leads to fast overfitting on the closed set speakers, and extractors obtained this way generalize badly for open-set task.

According to our observations, one reason of the x-vector success is the natural-gradient (NG) modification [30] of the stochastic gradient descend (SGD) optimization. The natural-gradient SGD procedure uses an inverse Fisher matrix for gradient scaling during training and prevents convergence to local optima.

We found out that the choice of the optimization objective is essential for training a discriminative speaker recognition system. According to our preliminary experiment results, angular margin softmax loss is much more effective than regular softmax loss as a discriminative objective. We did not try to apply A-softmax loss for x-vector extractor training but we believe that it can improve discriminative properties of the x-vectors.

Our studies also allow us to conclude that the performance of x-vector based systems can be improved by using cosine similarity metric learning method as a backend.

Another major issue of the speaker verification problem is the choice of the DNN architecture. The results presented in Tables 4, 5, 6, 7 demonstrate good performance of the proposed alternative extractors. Deep context based architecture SpeakerResNet with 24 layers is superior to SpeakerMaxPoolNet in real life conditions. We speculate that residual TDNN connections allow the network to automatically leverage necessary context for each level of feature abstraction.

Simple cosine similarity scoring method can be used for speaker verification in these systems. We've decided not to use CSML with SpeakerResNet and MaxPoolNet since A-softmax loss is specifically designed for use with cosine similarity and trains the last network layers accordingly.

When trained with augmented data, DNN-based speaker embedding systems significantly outperform our previous i-vector-based systems on SITW protocol [31].

8. Conclusion

This work demonstrates that DNN-based speaker embedding extractors can be effectively used for speaker verification.

- Choice of the optimization objective is essential for obtaining speaker embedding extractor with good generalization properties. To effectively discriminate speaker embeddings by cosine similarity a-softmax can be used.
- Speaker features are best captured with deep context. Using rather deep TDNN architectures with residual connections can capture speaker features from any level of feature abstraction and context.
- Performance of x-vector based systems could be improved by using cosine similarity metric learning approach for a backend model.

We proposed two speaker embedding extractors called SpeakerMaxPoolNet and SpeakerResNet which performed well during evaluation on SITW and NIST SRE 2016 corpora.

9. Acknowledgements

This work was financially supported by the Ministry of Education and Science of the Russian Federation, Contract 14.578.21.0189 (ID RFMEFI57816X0189).

10. References

[1] 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*, vol. 19, no. 4, pp. 788–798, 2011.

[2] Yun Lei, Nicolas Scheffer, Luciana Ferrer, and Mitchell McLaren, "A novel scheme for speaker recognition using a phonetically-aware deep neural network," in *2014 IEEE ICASSP*, pp. 1695–1699.

[3] Sergey Novoselov, Timur Pekhovsky, Oleg Kudashev, Valentin S Mendelev, and Alexey Prudnikov, "Non-linear PLDA for i-vector speaker verification," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[4] Oleg Kudashev, Sergey Novoselov, Timur Pekhovsky, Konstantin Simonchik, and Galina Lavrentyeva, "Usage of DNN in speaker recognition: advantages and problems," in *ISNN*. Springer, 2016, pp. 82–91.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[6] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, Andrew Rabinovich, et al., "Going deeper with convolutions," Cvpr, 2015.

[7] Diederik P Kingma and Max Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[8] Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song, "Sphereface: Deep hypersphere embedding for face recognition," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, vol. 1.

[9] Shi-Xiong Zhang, Zhuo Chen, Yong Zhao, Jinyu Li, and Yifan Gong, "End-to-end attention based text-dependent speaker verification," in *Spoken Language Technology Workshop (SLT), 2016 IEEE*. IEEE, 2016, pp. 171–178.

[10] Georg Heigold, Ignacio Moreno, Samy Bengio, and Noam Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5115–5119.

[11] Gautam Bhattacharya, Jahangir Alam, Themos Stafylakis, and Patrick Kenny, "Deep neural network based text-dependent speaker recognition: Preliminary results," *Odyssey 2016*, pp. 9–15, 2016.

[12] Ehsan Variani, Xin Lei, Erik McDermott, Ignacio Lopez Moreno, and Javier Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE ICASSP*. IEEE, 2014, pp. 4052–4056.

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

[14] Vijayaditya Peddinti, Daniel Povey, and Sanjeev Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[15] Hieu V. Nguyen and Li Bai, "Cosine similarity metric learning for face verification," in *Proceedings of the 10th Asian Conference on Computer Vision - Volume Part II*, Berlin, Heidelberg, 2011, ACCV'10, pp. 709–720, Springer-Verlag.

[16] Seyed Omid Sadjadi, Sriram Ganapathy, and Jason W Pelecanos, "The ibm 2016 speaker recognition system," *arXiv preprint arXiv:1602.07291*, 2016.

[17] Mitchell McLaren, Yun Lei, and Luciana Ferrer, "Advances in deep neural network approaches to speaker recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*. IEEE, 2015, pp. 4814–4818.

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

[19] David Snyder, Daniel Garcia-Romero, Gregory Sell, Daniel Povey, and Sanjeev Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," .

[20] Yandong Wen, Kaipeng Zhang, Zhifeng Li, and Yu Qiao, "A discriminative feature learning approach for deep face recognition," in *Computer Vision ECCV 2016: 14th European Conference, Part VII*, Amsterdam, The Netherlands, October 2016, pp. 499–515.

[21] Sumit Chopra, Raia Hadsell, and Yann LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. IEEE, 2005, vol. 1, pp. 539–546.

[22] Elad Hoffer and Nir Ailon, "Deep metric learning using triplet network," in *International Workshop on Similarity-Based Pattern Recognition*. Springer, 2015, pp. 84–92.

[23] Florian Schroff, Dmitry Kalenichenko, and James Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1026–1034.

[25] Galina Lavrentyeva, Sergey Novoselov, Egor Malykh, Alexander Kozlov, Oleg Kudashev, and Vadim Shchemelinin, "Audio replay attack detection with deep learning frameworks," *Proc. Interspeech 2017*, pp. 82–86, 2017.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[27] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al., "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*. IEEE Signal Processing Society, 2011, number EPFL-CONF-192584.

[28] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

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

[30] Daniel Povey, Xiaohui Zhang, and Sanjeev Khudanpur, "Parallel training of dnns with natural gradient and parameter averaging," *arXiv preprint arXiv:1410.7455*, 2014.

[31] Oleg Kudashev, Sergey Novoselov, Konstantin Simonchik, and Alexander Kozlov, "A speaker recognition system for the sitw challenge.," in *INTERSPEECH*, 2016, pp. 833–837.