MULTI-LEVEL DEEP NEURAL NETWORK ADAPTATION FOR SPEAKER VERIFICATION USING MMD AND CONSISTENCY REGULARIZATION

Weiwei Lin[1,2]*, Man-Wai Mak*[1]*, Na Li*[2]*, Dan Su*[2] *and Dong Yu*[2]

[1]Dept. of Electronic and Information Engineering
The Hong Kong Polytechnic University, Hong Kong, China
[2]Tencent AI Lab, China

ABSTRACT

Adapting speaker verification (SV) systems to a new environment is a very challenging task. Current adaptation methods in SV mainly focus on the backend, i.e, adaptation is carried out after the speaker embeddings have been created. In this paper, we present a DNN-based adaptation method using maximum mean discrepancy (MMD). Our method exploits two important aspects neglected by previous research. First, instead of minimizing domain discrepancy at utterance-level alone, our method minimizes domain discrepancy at both frame-level and utterance-level, which we believe will make the adaptation more robust to the duration discrepancy between training data and test data. Second, we introduce a consistency regularization for unlabelled target-domain data. The consistency regularization encourages the target speaker embeddings robust to adverse perturbations. Experiments on NIST SRE 2016 and 2018 show that our DNN adaptation works significantly better than the previously proposed DNN adaptation methods. What's more, our method works well with backend adaptation. By combining the proposed method with backend adaptation, we achieve a 9% improvement over backend adaptation in SRE18.

Index Terms— Speaker verification;domain adaptation;data augmentation;maximum mean discrepancy;transfer learning;

1. INTRODUCTION

A fundamental assumption of machine learning is that training data and test data are sampled from the same underlying distribution [1, 2]. In practice, a lot of factors can undermine this assumption. This is especially the case when we want to deploy a model to a new environment, where the data have different properties than the training data. For speaker verification, this could happen when the new environment has some specific noise and channel conditions or involves speakers speaking different languages than the training speakers. Directly using models in these situations could result in poor performance. Fortunately, it is often possible to collect a small amount of data from the new environment. These data are typically referred to as target-domain data in the literature [3]. The other data are referred to as source-domain data. The process of adapting a model to the production environment is referred to as domain adaptation (DA).

The domain mismatch investigated in this paper is language mismatch. In NIST speaker recognition evaluation (SRE) 2016 [4], the language mismatch problem was brought to SV researchers for the first time.

State-of-art SV systems are comprised of a deep neural network and a backend model [5]. DA is typically carried out in the backend. In Kaldi's SRE16 recipe, adaptation is carried out in the PLDA model's mean and covariance matrix. It has been shown to be very effective and adopted by many researchers [6, 7]. Another very popular DA method in the backend is correlation alignment (CORAL), which essentially whitens the source-domain data and recolors them with a whitening matrix estimated from target-domain data [8]. In [9], the author proposed a hybrid method combining PLDA model adaptation and CORAL and showed that it is superior to the individual methods. A more complicated backend adaptation were proposed in [10, 11]. The authors proposed to used an auto-encoder to minimize the maximum mean discrepancy between source-domain data and target-domain data. The method can also address multi-source domain mismatch.

DNN adaptation is relatively new in SV. Because DNN provides a larger parameter space to explore, it is potentially more powerful than backend adaptation. In [12], the authors proposed to use adversarial learning to adapt the speaker embeddings. Specifically, Wasserstein GANs were used to minimize the discrepancy between the source-domain and the target-domain speaker embeddings. The authors also explored using other information like language labels and phone numbers and found that they are beneficial. However, their method requires speaker labels to perform well, which limits the method's applicability. In [13], several GAN variants based on the similar idea [12] were proposed. Both adapta-

Part of this work was done during Weiwei Lin's internship at Tencent AI Lab. This work was supported by RGC of Hong Kong, Grant No. 152137/17E and Tencent AI Lab Rhino-Bird Gift Fund.

tion and verification were carried out end-to-end. However, the performance of the system is not as good as the x-vector system with PLDA adaptation in Kaldi.

2. MAXIMUM MEAN DISCREPANCY

Maximum mean discrepancy is a distance measure on the space of probability [14]. Given two sets of samples $\mathcal{X} = \{\mathbf{x}_i\}_{i=1}^M$ and $\mathcal{Y} = \{\mathbf{y}_j\}_{j=1}^N$ from distributions P_x and P_y, MMD measures the similarity of P_x and P_y by computing the mean squared difference of the statistics of the samples:

$$\mathcal{D}(\mathcal{X}, \mathcal{Y}) = \left\| \frac{1}{N} \sum_{i=1}^{N} \phi(\mathbf{x}_i) - \frac{1}{M} \sum_{j=1}^{M} \phi(\mathbf{y}_j) \right\|^2. \quad (1)$$

When $\phi(.)$ is the identity function, the MMD computes the mean squared distance between the sample sets. Eq. 1 can be expanded as:

$$\mathcal{D}(\mathcal{X}, \mathcal{Y}) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \phi(\mathbf{x}_i)^\mathsf{T} \phi(\mathbf{x}_{i'}) - \frac{2}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \phi(\mathbf{x}_i)^\mathsf{T} \phi(\mathbf{y}_j) + \frac{1}{M^2} \sum_{j=1}^{M} \sum_{j'=1}^{M} \phi(\mathbf{y}_j)^\mathsf{T} \phi(\mathbf{y}_{j'}). \quad (2)$$

The dot product terms can be replaced with kernel functions $k(\cdot, \cdot)$:

$$\mathcal{D}(\mathcal{X}, \mathcal{Y}) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} k(\mathbf{x}_i, \mathbf{x}_{i'}) - \frac{2}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} k(\mathbf{x}_i, \mathbf{y}_j) + \frac{1}{M^2} \sum_{j=1}^{M} \sum_{j'=1}^{M} k(\mathbf{y}_j, \mathbf{y}_{j'}). \quad (3)$$

Another popular kernel is the radial basis function (RBF) kernel:

$$k(\mathbf{x}, \mathbf{y}) = \exp\left(-\frac{1}{2\sigma^2} \|\mathbf{x} - \mathbf{y}\|^2 \right), \quad (4)$$

where σ is the width parameter. With the RBF kernel, the feature space is of infinite dimension and contains all moments of data. Minimizing MMD using the RBF kernel is equivalent to matching all moments of two distributions [15].

3. DNN DOMAIN ADAPTATION

3.1. Network Architecture

We modify Kaldi's x-vector architecture by replacing TDNN with CNN and add two convolutional layers with a stride 2 and a kernel size of 2. The statistics pooling layer and the last two fully-connected layers are the same as the x-vectors network [5, 16]. Table 1 summarizes the architecture of our network.

Table 1. Summary of our neural network architecture. The kernel is specified as kernel_size, stride, and dilation

Layer	Kernel	Channel_in × Channel_out
Conv1	5,1,1	23 × 512
Conv2	3,1,2	512 × 512
Conv3	3,1,3	512 × 512
Conv4	1,1,1	512 × 512
Conv5	1,1,1	512 × 1536
Statistics pooling		1536 × 3072
FC6	–	3072 × 512
FC7	–	512 × 512
AM-softmax	–	512 × N

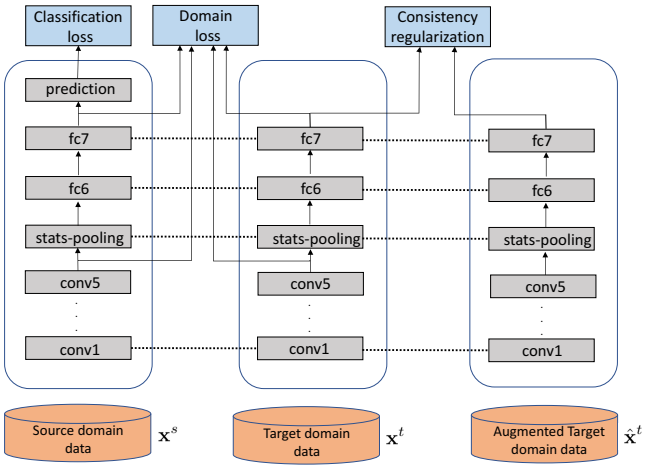

Fig. 1. The architecture of our proposed method. The network is trained to minimize the classification loss and the domain loss with consistency regularization (see Eq. 9). For target-domain data, no label is required. The dotted lines indicate weight-sharing within individual layers.

3.2. Multi-level Adaptation

Assume that we have a labeled dataset $\{(\mathbf{x}_i^s, y_i^s)\}_{i=1}^I$ from the source-domain and let $\Theta = \{\mathbf{W}^l, \mathbf{b}^l\}_{l=1}^L$ denotes the set of all network parameters. The objective function of the network can be written as:

$$\min_\Theta \frac{1}{I} \sum_{i=1}^{I} J(p_\Theta(y|\mathbf{x}_i^s), y_i^s), \quad (5)$$

where J is the cross-entropy function and $p_\Theta(y|\mathbf{x}_i^s)$ is the conditional probability that the network assigns $\mathbf{x}_i^s$ to class y. Minimizing this objective alone will not guarantee the generalization to the target-domain. To make generalization to the target-domain possible, we need to reduce the divergence between the marginal distribution of the source-domain and the target domain. In neural networks, we typically reduce the divergence in the hidden activations. Let $\mathcal{H}_*^l = \{\mathbf{h}_i^l\}$ denotes

the l-th layer hidden activations for source or target data. The cross-entropy loss together with MMD distance is

$$\min_{\Theta} \frac{1}{I} \sum_{i=1}^{I} J(p_{\Theta}(y|\mathbf{x}_i^s), y_i^s) + \lambda \cdot \mathcal{D}(\mathcal{H}_s^l, \mathcal{H}_t^l), \quad (6)$$

where λ is a constant controlling the trade-off between the two objectives.

As mentioned in [17], deeper layers typically have larger domain discrepancy gaps. Therefore, it is very common to minimize the divergence at the network's last layer. In our case, it is the 7-th layer, i.e., $l = 7$. However, the current DNN training scheme typically uses very short speech segments (200 frames to 400 frames) for training and relies on the backend to compensate for the duration discrepancy. The embedding distribution may shift with speech duration, which may result in an inaccurate divergence estimate. Adapting frame-level activations, on the other hand, has no such problem. Therefore, we argue that it is important to adapt frame-level features as well. Here we choose the last convolutional layer before statistics pooling, i.e., $l = 5$ for adaptation:

$$\min_{\Theta} \frac{1}{I} \sum_{i=1}^{I} J(p_{\Theta}(y|\mathbf{x}_i^s), y_i^s) + \lambda \cdot \mathcal{D}(\mathcal{H}_s^7, \mathcal{H}_t^7) + \alpha \cdot \mathcal{D}(\mathcal{H}_s^5, \mathcal{H}_t^5)). \quad (7)$$

3.3. Consistency Training

Data augmentation is the most important part of x-vector's success. However, how to use data augmentation on unlabelled data have not been explored in SV. Consistency training has been successfully explored in semi-supervised learning [18]. The idea is to enforce or regularize a network such that the network predictions are consistent even if the network's input is subject to noise perturbation. In [18], the regularization is achieved by minimizing the following KL divergence:

$$\mathop{\mathbb{E}}_{q(\hat{\mathbf{x}}|\mathbf{x})} [\mathrm{KL}(p_{\Theta}(y|\mathbf{x})||p_{\Theta}(y|\hat{\mathbf{x}}))], \quad (8)$$

where $\mathbf{x}$ denotes the original data, $\hat{\mathbf{x}}$ denotes the augmented data, and $q(\hat{\mathbf{x}}|\mathbf{x})$ is a data augmentation transformation defining the noise added process. Note that Eq 8 requires labels or hypothesized labels. We propose another form of consistency penalty. First, instead of minimizing the KL divergence between the predictions conditioned on original data and predictions conditioned on augmented data. We propose minimizing the discrepancy between the embedding produced by the original data and embedding produced by the augmented data. The motivation is that DNN embedding should be robust to input perturbation. After all, the goal of DNN is to create speaker embedding instead of prediction. Secondly, instead of using KL divergence, we use MMD to measure consistency. Let $\mathcal{H}_t^7$ and $\hat{\mathcal{H}}_t^7$ denotes the set of original data embeddings and augmented data embeddings, respectively. The consistency regularization using MMD can be written as:

$$\mathcal{D}(\mathcal{H}_t^7, \hat{\mathcal{H}}_t^7) = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} k(\mathbf{h}_i^7, \mathbf{h}_{i'}^7) - \frac{2}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} k(\mathbf{h}_i^7, \hat{\mathbf{h}}_j^7) + \frac{1}{M^2} \sum_{j=1}^{M} \sum_{j'=1}^{M} k(\hat{\mathbf{h}}_j^7, \hat{\mathbf{h}}_{j'}^7). \quad (9)$$

By combining the consistency regularization with Eq. 8 we have the total loss function:

$$\min_{\Theta} \frac{1}{I} \sum_{i=1}^{I} J(\theta(\mathbf{x}_i^s), y_i^s). + \lambda \cdot \mathcal{D}(\mathcal{H}_s^7, \mathcal{H}_t^7) + \alpha \cdot \mathcal{D}(\mathcal{H}_s^5, \mathcal{H}_t^5)) + \beta \cdot \mathcal{D}(\mathcal{H}_t^7, \hat{\mathcal{H}}_t^7). \quad (10)$$

Figure 1 summarizes the architecture and objective functions.

4. EXPERIMENTS

4.1. Data Preparation

The training data include NIST SRE 2004–2010 (SRE04-10 in short) and all of the Switchboard data. We follow the data augmentation strategy in Kaldi SRE16 receipt. The training data were augmented by adding noise, music, reverb, and babble to the original speech files in the datasets. After filtering out utterances shorter than 500 frames and speakers with less than 8 utterances, we are left with 4808 speakers. 23-dimensional Mel-frequency cepstral coefficients (MFCC) were computed from 8kHz speech files. Mean normalization was applied to the MFCC using a 3-second sliding window. Non-speech frames were removed using Kaldi's energy-based voice activity detector.

4.2. DNN and Backend Training

The value of λ, β and α were all set to 1. For the Gaussian bandwidth σ, we followed the median heuristics and estimated median pairwise distance σ_m from the training data [14]. A total of 19 Gaussian kernels were used with varying bandwidth σ between $2^{-8}\sigma_m$ and $2^{8}\sigma_m$ with a multiplicative step-size of $2^{0.5}$. All DNNs were trained using a batch size of 32 and were optimized by the Adam optimizer [19] with a learning rate of 0.001. The networks were implemented in Pytorch [20]. We used correlation alignment [8] for domain adaptation in the PLDA backend.

4.3. Evaluation

All systems were evaluated on the evaluation set of SRE 2016 and 2018. The SRE16 evaluation set is composed of Tagalog and Cantonese telephone conversations. For SRE18, we only conducted the evaluation on the CMN2 portion, which consists of Tunisian Arabic conversations. We report results in

terms of equal error rate (EER) and minimum cost function (DCF). Both metrics were obtained using the scoring tools provided by NIST.

5. RESULTS

Table 2. Comparison with other DNN adaptation methods. Sup. WGAN [12] used the labels of SRE16 and SRE18 development data. There is no backend adaptation in all of the systems.

	SRE16		SRE18	
Adapt Method	EER (%)	minDCF	EER(%)	minDCF
WGAN [12]	13.25	0.899	9.59	0.652
Sup. WGAN [12]	9.59	0.746	8.88	0.619
LSGAN [21]	11.74	-	-	-
Our DNN Adapt.	**9.03**	**0.585**	**8.33**	**0.520**

Table 3. The Performances of CORAL, PLDA adaptation and the proposed method.

	SRE16		SRE18	
Adapt Method	EER(%)	minDCF	EER(%)	minDCF
Our DNN Adapt.	9.03	0.585	8.33	0.520
CORAL Adapt.	8.49	0.560	8.74	0.553
PLDA Adapt.	8.55	0.556	8.88	0.563
Ours+CORAL Adapt.	**8.28**	**0.541**	8.13	0.519
Ours+PLDA Adapt.	8.29	0.546	**8.09**	**0.521**

Table 4. Ablation study of multi-level adaptation and consistency regularization in the proposed method.

			SRE16		SRE18	
Layer 7	Layer 6	Consis.	EER(%)	DCF	EER(%)	DCF
×	×	×	12.02	0.990	11.59	0.72
✓	×	×	9.79	0.621	9.08	0.580
✓	✓	×	9.63	0.606	8.77	0.555
✓	✓	✓	9.03	0.585	8.33	0.520

5.1. Comparison with Other DNN Adaptations

In this section, we compare the proposed method with the previously proposed DNN adaptation methods. The latter includes Wasserstein GAN (WGAN) adaptation, supervised WGAN adaptation in [12] and least square GAN (LSGAN) in [21]. The results are presented in Table 2. All the results in Table 2 are without additional backend adaptation. It is clear from the table that our method performs significantly better than the previously proposed methods. It is worth noting that our method even performs better than the supervised adaptation in [12].

5.2. Comparison with Backend Adaptations

In this section, we compare the proposed method with two popular backend adaptation methods, namely, CORAL [8] and Kaldi's PLDA adaptation [5]. The potential of combining the proposed method and backend adaptation is also investigated. The results are presented in Table 3. As can be seen from Table 3, in SRE16, CORAL is the most effective adaptation method. However, in SRE18, our method has a clear advantage over the backend adaptations. This, we believe, is due to the fact that SRE18 has more data for adaptation (over 4000 utterances compared with only 2340). Besides, it seems that the proposed method works well with backend adaptation, as combining them improves performance.

5.3. Ablation Study

To investigate whether multi-level adaptation and consistency regularization are effective, we also carried out an ablation study. The results are presented in Table 4. We can see that adapting the 7-th layer alone already gives great improvement over no adaptation. Adapting the 6-th layer gives a small performance gain for both. Consistency regularization also improves the performance in both SRE16 and SRE18.

6. CONCLUSIONS

In this paper, we proposed a DNN domain adaptation using maximum mean discrepancy and consistency regularization. The proposed method significantly outperforms the previously proposed DNN adaptation and works well with backend adaptations. However, we only investigate the language induced domain difference. It would be interesting to compare other factors such as noise and channel induced domain differences in the future.

7. REFERENCES

[1] S. B. David, T. Lu, T. Luu, and D. Pál, "Impossibility theorems for domain adaptation," in *Proc. the Thirteenth International Conference on Artificial Intelligence and Statistics*, pp. 129–136, 2010.

[2] Y. Mansour, M. Mohri, and A. Rostamizadeh, "Domain adaptation: Learning bounds and algorithms," *arXiv preprint arXiv:0902.3430*, 2009.

[3] G. Csurka, "Domain adaptation for visual applications: A comprehensive survey," *arXiv preprint arXiv:1702.05374*, 2017.

[4] S. O. Sadjadi, T. Kheyrkhah, A. Tong, C. Greenberg, D. Reynolds, E. Singer, L. Mason, and J. Hernandez-Cordero, "The 2016 NIST speaker recognition evaluation," in *Proc. Interspeech*, pp. 1353–1357, 2017.

[5] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust DNN embeddings for speaker recognition," in *Proc. ICASSP*, pp. 5329–5333, IEEE, 2018.

[6] K. A. Lee and S. I. Group, "The I4U mega fusion and collaboration for NIST speaker recognition evaluation 2016," in *Proc. Interspeech*, pp. 1328–1332, 2017.

[7] O. Plchot, P. Matějka, A. Silnova, O. Novotný, M. D. Sánchez, J. Rohdin, O. Glembek, N. Brümmer, A. Swart, J. Jorrín-Prieto, P. García, L. Buera, P. Kenny, J. Alam, and G. Bhattacharya, "Analysis and description of ABC submission to NIST SRE 2016," in *Proc. Interspeech*, pp. 1348–1352, 2017.

[8] B. Sun, J. Feng, and K. Saenko, "Correlation alignment for unsupervised domain adaptation," in *Domain Adaptation in Computer Vision Applications*, pp. 153–171, Springer, 2017.

[9] K. A. Lee, Q. Wang, and T. Koshinaka, "The CORAL+ algorithm for unsupervised domain adaptation of plda," in *Proc. ICASSP*, pp. 5821–5825, IEEE, 2019.

[10] W. W. Lin, M. W. Mak, L. X. Li, and J. T. Chien, "Reducing domain mismatch by maximum mean discrepancy based autoencoders.," in *Odyssey*, pp. 162–167, 2018.

[11] W. W. Lin, M. W. Mak, and J. T. Chien, "Multisource i-vectors domain adaptation using maximum mean discrepancy based autoencoders," *IEEE/ACM Transactions on Audio, Speech and Language Processing*, vol. 26, no. 12, pp. 2412–2422, 2018.

[12] J. Rohdin, T. Stafylakis, A. Silnova, H. Zeinali, L. Burget, and O. Plchot, "Speaker verification using end-to-end adversarial language adaptation," in *Proc. ICASSP*, pp. 6006–6010, 2019.

[13] G. Bhattacharya, J. Monteiro, J. Alam, and P. Kenny, "Generative adversarial speaker embedding networks for domain robust end-to-end speaker verification," in *Proc. ICASSP*, pp. 6226–6230, 2019.

[14] A. Gretton, K. M. Borgwardt, M. Rasch, B. Schölkopf, and A. J. Smola, "A kernel method for the two-sample-problem," in *Proc. Advances in Neural Information Processing Systems*, pp. 513–520, 2007.

[15] Y. Li, K. Swersky, and R. Zemel, "Generative moment matching networks," in *Proc. International Conference on Machine Learning*, pp. 1718–1727, 2015.

[16] V. Peddinti, D. Povey, and S. Khudanpur, "A time delay neural network architecture for efficient modeling of long temporal contexts," in *Proc. Interspeech*, 2015.

[17] M. Long, Y. Cao, J. Wang, and M. Jordan, "Learning transferable features with deep adaptation networks," in *Proc. International Conference on Machine Learning*, pp. 97–105, 2015.

[18] Q. Xie, Z. Dai, E. Hovy, M. T. Luong, and Q. V. Le, "Unsupervised data augmentation," *arXiv preprint arXiv:1904.12848*, 2019.

[19] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[20] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, pp. 8024–8035, 2019.

[21] G. Bhattacharya, J. Alam, and P. Kenny, "Adapting end-to-end neural speaker verification to new languages and recording conditions with adversarial training," in *Proc. ICASSP*, pp. 6041–6045, IEEE, 2019.