VAE-based regularization for deep speaker embedding

Yang Zhang[1,2], *Lantian Li*[1], *Dong Wang*[1*]

[1]Center for Speech and Language Technologies, Tsinghua University, China
[2]Beijing University of Posts and Telecommunications, China

{zhangyang,lilt}@cslt.org; wangdong99@mails.tsinghua.edu.cn

Abstract

Deep speaker embedding has achieved state-of-the-art performance in speaker recognition. A potential problem of these embedded vectors (called 'x-vectors') are not Gaussian, causing performance degradation with the famous PLDA back-end scoring. In this paper, we propose a regularization approach based on Variational Auto-Encoder (VAE). This model transforms x-vectors to a latent space where mapped latent codes are more Gaussian, hence more suitable for PLDA scoring.

Index Terms: Variational Auto-Encoder, deep speaker embedding

1. Introduction

Automatic speaker verification (ASV) has found a broad range of applications. Conventional ASV methods are based on statistical models [1, 2, 3]. Perhaps the most famous statistical model in ASV is the Gaussian mixture model-universal background model (GMM-UBM) [1]. This model represents the 'main' variance of speech signals by a set of global Gaussian components (UBM), and the speaker characters are represented as the 'shift' of speaker-dependent GMMs over each Gaussian component of the UBM, denoted by a 'speaker supervector'. The GMM-UBM architecture was later enhanced by subspace models, which assume that a speaker supervector can be factorized into a speaker vector (usually low-dimensional) and a residual that represents intra-speaker variation. The joint factor analysis [2, 4] was the most successful subspace model in early days, though the following i-vector model obtained more attention [3]. Besides the simple structure and the superior performance, the i-vector approach firstly demonstrated that a speaker can be represented by a low-dimensional vector, which is the precursor of the important concept of **speaker embedding**.

It should be emphasized, however, that the i-vector model is purely unsupervised and the embeddings (i-vectors) contain a multitude of variations more than speaker information. Therefore, it heavily relies on a powerful back-end scoring model to achieve reasonable performance. Among various back-end models, the PLDA model [5, 6] has been very powerful, in particular with a simple whitening and length normalization [7]. In the nut shell, PLDA assumes the 'true' speaker codes within an i-vector is low dimensional and follows a simple Gaussian prior, and the residual is a full-rank Gaussian, formally written by:

$$\phi_{su} = m + \mathrm{U}y_s + \epsilon_{su}, \tag{1}$$

where ϕ_{su} is the i-vector of utterance u of speaker s, $y_s \sim N(0, I)$ and $\epsilon_{su} \sim N(0, W)$ are speaker codes and residual respectively, m is the global shift and U is the speaker loading matrix. Under this assumption, the speaker prior $p(y_s)$, the conditional $p(\phi|y_s)$ and the marginal $p(\phi)$ are all Gaussian. Fortunately, i-vectors match these conditions pretty well, due to the linear Gaussian structure of the i-vector model. Partly for this reason, the i-vector/PLDA framework remains a strong baseline on many ASV tasks.

Recently, neural-based ASV models have shown great potential [8, 9, 10, 11]. These models utilize the power of deep neural networks (DNNs) to learn strong speaker-dependent features, ideally from a large amount of speaker-labelled data. The present research can be categorized into frame-based learning [8, 10] and utterance-based learning [9, 11, 12, 13]. The frame-based learning intends to learn short-time speaker features, thus more generally useful for speaker-related tasks, while the utterance-based learning focuses on a whole-utterance speaker representation and/or classification, hence more suitable for the ASV task. A popular utterance-based learning approach is the x-vector model proposed by Snyder et al. [11], where the first- and second-order statistics of frame-level features are collected and projected to a low-dimensional representation called x-vector, with the objective of discriminating the speakers in the training dataset. The x-vector model has achieved good performance in various speaker recognition tasks, as well as related tasks such as language identification [14]. Essentially, the x-vector model can be regarded as a deep and discriminative counterpart of the i-vector model, and is often called **deep speaker embedding**.

Interestingly, experiments show that the x-vector system also heavily relies on a strong back-end scoring model, in particular PLDA. Since the x-vector have been sufficiently discriminative, the role of PLDA here is **regularization** rather than **discrimination** (as in the i-vector paradigm): it (globally) discovers the underlying speech codes that are intrinsically Gaussian, so that the ASV scoring based on these codes tends to be comparable across speakers. A potential problem, however, is that x-vectors inferred from DNNs are unconstrained, which means that the speaker distribution and the speaker conditional could be in any form. These unconstrained distributions may cause great difficulty for PLDA to discover the underlying speaker codes that are assumed to be Gaussian. Some researchers have noticed this problem and proposed some remedies that encourage speaker conditionals more Gaussian [15, 16], but none of them constrain the prior, thus produced x-vectors are still not suitable for PLDA modeling.

In this paper, we investigate an explicit regularization model for unconstrained x-vectors. This model is inspired by the variational auto-encoder (VAE) architecture, which is capable of projecting an unconstrained distribution to a simple Gaussian distribution. This can be used to constrain the marginal distribution of x-vectors. Moreover, a cohesive loss is added to the VAE objective. This follows the same spirit of [15, 16] and can constrain the speaker conditionals. Experiments showed that with this VAE-based regularization, performance of cosine scoring is largely improved, even comparable with PLDA. This indicates that VAE plays a similar role as PLDA, or, in other words, PLDA works as a regularizer rather than a discriminator in the x-vector scoring. Furthermore, the VAE-based speaker codes achieved the state-of-the-art performance when scoring with PLDA, demonstrating that (1) VAE-based speaker codes are more regularized and suitable for PLDA modeling, and (2) VAE-based regularization and PLDA scoring are complementary.

The organization of this paper is as follows. Section 2 presents the VAE-based regularization model, and the experiments are reported in Section 3. The paper is concluded in Section 4.

2. VAE-based speaker regularization

2.1. Revisit PLDA

The principle of PLDA is to model the marginal distribution of speaker embeddings (i-vector or x-vector), by factoring the total variation of the embeddings into between-speaker variation and within-speaker variation. Based on this factorization, the ASV decision can be cast to a hypothesis test [5, 6], formulated by:

$$\begin{aligned} s(\phi_1,\phi_2) &= \frac{P(\phi_1=\phi_2|\Lambda)}{P(\phi_1\neq\phi_2|\Lambda)} \\ &= \frac{\int p(\phi_1,\phi_2|y)p(y)\mathrm{d}y}{\int p(\phi_1|y)p(y)\mathrm{d}y\int p(\phi_2|y)p(y)\mathrm{d}y}, \end{aligned}$$

where s denotes the confidence score, and the equality relation ($\phi_1=\phi_2$) denotes that the two embeddings are from the same speaker.

According to Eq.(1), PLDA is a linear Gaussian model and the prior, the conditional, and the marginal are Gaussian. If the embeddings do not satisfy this condition, PLDA cannot model them well, leading to inferior performance. This is the case of x-vectors, which are derived from DNNs and both the speaker prior and speaker conditionals are unconstrained. In order to deal with the unconstrained distributions of x-vectors, we need a probabilistic model more complex than PLDA.

2.2. VAE for regularization

VAE is a generative model (like PLDA) that can represent a complex data distribution [17]. The key idea of VAE is to learn a DNN-based mapping function $x=f(z)$ that maps a simple distribution $p(z)$ to a complex distribution $p(x)$. In other words, it represents complex observations by simple-distributed latent codes via **distribution mapping**. An illustration of this mapping is shown in Fig. 1. It can be easily shown that the mapped distribution is written by:

$$\log p(x)=\log p(z)+\log|\det\frac{\mathrm{d}f^{-1}(x)}{\mathrm{d}x}|,$$

where f^{-1} is the inverse function of $f(z)$.

Although VAE can be used to represent the complex **marginals**, it does not involve any class structure, and so cannot be used in the hypothesis test scoring framework. Nevertheless, if we can find the posterior $p(z|x)$, the complex $p(x)$ can be mapped to a more constrained $p(z)$, so the simple cosine distance can be used for verification. Moreover, the regularized code z tends to be easily modeled by PLDA, hence combining the strength of VAE in distribution mapping and the strength of PLDA in distinguishing between- and within-speaker variations. Fortunately, VAE provides a simple way to infer an approximation distribution of $p(z|x)$, denoted by $q(z|x)$. It learns a function $g(x)$, parameterized by a DNN, to map x to the parameters of $q(z|x)$, which are the mean and covariance if $q(z|x)$ is assumed to be Gaussian. By this setting, the mean vector of $q(z|x)$ can be treated as VAE-regularized speaker codes, and can be used in cosine or PLDA-based scoring.

Fig. 2 illustrates the VAE framework. In this framework, a **decoder** $f(z)$ maps $p(z)$ to $p(x)$, i.e.,

$$p(x)=\int p(x|z)p(z)\mathrm{d}z=\int N(f(z),I)p(z)\mathrm{d}z,$$

where $p(x|z)$ has been assumed to be a Gaussian. Furthermore, an **encoder** $g(x)$ produces a distribution $q(z|x)$ that approximates the posterior distribution $p(z|x)$ as follows:

$$p(z|x)\approx q(z|x)=N(\mu(x),\sigma(x)),$$

where $[\mu(x)\ \ \sigma(x)]=g(x)$.

The training objective is the log probability of the training data $\sum_i \ln p(x_i)$. It is intractable so a variational lower bound is optimized instead, which depends on both the encoder $g(x)$ and the decoder $f(z)$. This is formally written by:

$$L(f,g)=\sum_i\{-D_{\mathrm{KL}}[q(z|x_i)||p(z)]+\mathbb{E}_{q(z|x_i)}[\ln p(x_i|z)]\},$$

where D_{KL} is the KL distance, and $\mathbb{E}_q$ denotes expectation w.r.t. distribution q. As the expectation is intractable, a sampling scheme is often used, as shown in Fig. 2. More details of the training process can be found in [17].

Note that the $L(f,g)$ involves two components: a regularization term that pushes $q(z|x)$ to $p(z)$, and a reconstruction term that encourages a good reconstruction of x from z. We are free to tune the relative weights of these two terms in practice, in order to obtain latent codes that are either more regularized or more representative. This freely-modified objective may be never a variational lower bound, though non-balanced weights often lead to better performance in our experiments.

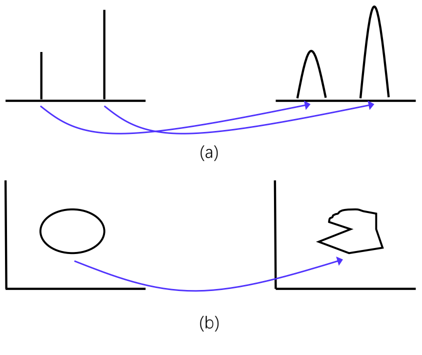

Figure 1: *Two examples of distribution mapping. (a) a discrete distribution is mapped to a mixture of two Gaussians; (b) a 2-dim Gaussian is mapped to an irregular distribution.*

2.3. Speaker cohesive VAE

The standard VAE only constrains the marginal distribution $p(z)$ to be Gaussian, which does not guarantee a Gaussian prior or a Gaussian conditional. This is because the VAE model is purely unsupervised and there is no speaker information involved. This lack of speaker information is probably not a big

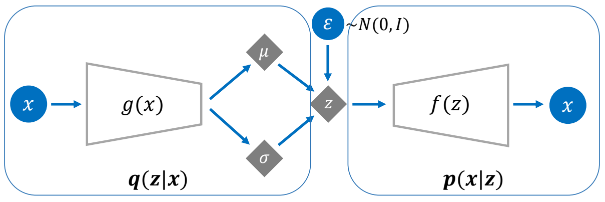

Figure 2: *The standard VAE architecture. It involves a generative model (decoder) $p(z)$ and $p(x|z)$, and an inference model (encoder) $q(z|x)$ that approximates the posterior $p(z|x)$. Both $p(x|z)$ and $q(z|x)$ involve a DNN-based mapping function. A random variable ϵ is used to facilitate the construction of $q(z|x)$, known as 'reparameterization trick' [17].*

issue for x-vectors as they are speaker discriminative already. However, considering speaker information may help VAE to produce a better regularization. Especially, if the speaker code $z_{s,u}$ of a particular speaker s can be regularized to be Gaussian, the scores based on either cosine distance or PLDA will be more across-speaker comparable. This can be formulated as an additional term in the VAE objective function, which we call **speaker cohesive loss** denoted by L_{C}:

$$L_{\mathrm{C}}(f,g) = \sum_i \ln p(\mu(x)|s(x)) = \sum_i \ln N(\mu(x); s(x), I)$$

where $s(x)$ denotes the mean of $\mu(x)$ of all utterances that belong to the same speaker as x. This essentially follows the same spirit of the central loss used in [15, 16]. Fig. 3 illustrates the VAE architecture with cohesive loss, and we name this improved VAE architecture as **Cohesive VAE**.

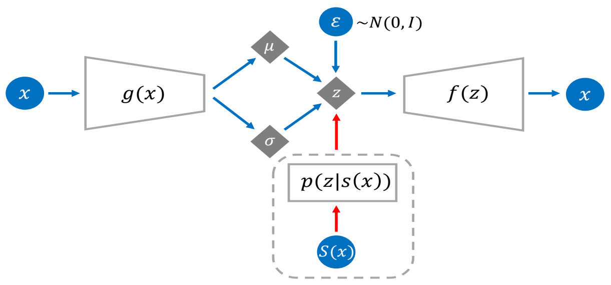

Figure 3: *The VAE architecture with cohesive loss (dotted box).*

3. Experiments

3.1. Data

Three datasets were used in our experiments: VoxCeleb, SITW and CSLT-SITW. VoxCeleb was used for model training, while the other two were used for evaluation. More information about these three datasets is presented below.

VoxCeleb: A large-scale free speaker database collected by University of Oxford, UK [18]. The entire database involves *VoxCeleb1* and *VoxCeleb2*. This dataset, after removing the utterances shared by SITW, was used to train the x-vector model, plus the PLDA and VAE models. Data augmentation was applied, where the MUSAN corpus [19] was used to generate noisy utterances and the room impulse responses (RIRS) corpus [20] was used to generate reverberant utterances.

SITW: A standard database used to test ASV performance in real-world conditions [21]. It was collected from open-source media channels, and consists of speech data covering 299 well-known persons. There are two standard datasets for testing: *Dev. Core* and *Eval. Core*. We used *Dev. Core* to select model parameters, and *Eval. Core* to perform test in our first experiment. Note that the acoustic condition of SITW is similar to that of the training set VoxCeleb, so this test can be regarded as an **in-domain test**.

CSLT-SITW: A small dataset collected by CSLT for commercial usage. It consists of 77 speakers, each of which records a simple Chinese command word, and the duration is about 2 seconds. The scenarios involve laboratory, corridor, street, restaurant, bus, subway, mall, home, etc. Speakers varied their poses during the recording, and the recording devices were placed both near and far. There are about $30k$ utterances in total. The acoustic condition of this dataset is quite different from that of the training set VoxCeleb, and was used for **out-of-domain test**.

3.2. Settings

We built several systems to validate the VAE-based regularization, each involving a particular pair of front-end and back-end.

3.2.1. Front-end

x-vector: The baseline x-vector front-end. It was built following the Kaldi SITW recipe [22]. The feature-learning component is a 5-layer time-delay neural network (TDNN). The statistic pooling layer computes the mean and standard deviation of the frame-level features from a speech segment. The size of the output layer is $7,185$, corresponding to the number of speakers in the training set. Once trained, the 512-dimensional activations of the penultimate hidden layer are read out as an x-vector.
v-vector: The VAE-regularized speaker code. The VAE model is a 7-layer DNN. The dimension of code layer is 200, and other hidden layers are $1,800$. The x-vectors of all the training utterances are used to the VAE training.
c-vector: The VAE-regularized speaker code, with the cohesive loss involved in the VAE training. The model structure is the same as in the v-vector front-end, and the v-vector VAE was used as the initial model for training. We tuned the weight $L_{\mathrm{C}}(f,g)$ in the objective function, and found 10 is a reasonable value.
a-vector: Speaker code regularized by a standard auto-encoder (AE). AE shares a similar structure as VAE, but the latent codes are not probabilistic so it is less capable of modeling complex distributions. The AE structure is identical to the VAE model in the v-vector front-end, except that the code layer is deterministic.

3.2.2. Back-end

Cosine: Simple cosine distance.
PCA: PCA-based projection (150-dim) plus cosine distance.
PLDA: PLDA scoring.
L-PLDA: LDA-based projection (150-dim) plus PLDA scoring.
P-PLDA: PCA-based projection (150-dim) plus PLDA scoring.

3.3. In-domain test

The results on the two SITW evaluation sets, *Dev. Core* and *Eval. Core*, are reported in Table 1. The results are reported in terms of equal error rate (EER).

Firstly focus on the x-vector front-end. It can be found that PLDA scoring outperformed cosine distance. As we argued, this cannot be interpreted as the discriminative nature of PLDA, but its regularization capability. This is supported by the observation that the v-vector front-end achieved rather good performance with the cosine back-end (compared with x-vector + PLDA). Since VAE is purely unsupervised, it only contributes to regularization. This suggests that PLDA may play a similar role as VAE.

Secondly, we observe that with PCA or LDA, PLDA can perform much better. It is not convincing to assume that LDA and PCA improve the discriminant power of x-vectors (in particular PCA), so the only interpretation is that these two models performed regularization, generating more Gaussian codes that are suitable for PLDA. This regularization is similar as what VAE did, but it seems VAE did a better job than PCA, and even better than LDA on the larger evaluation set *Eval. Core*, even without any speaker supervision.

Thirdly, it can be found that c-vectors performed better than v-vectors with cosine scoring, confirming that involving cohesive loss improves the regularization. When combined with PLDA, however, the advantage of c-vectors diminished. This is expected as PLDA has already learned the speaker discriminative knowledge.

Finally, we found that other unsupervised regularization methods, including PCA and AE, can not obtain reasonable performance with cosine distance, indicating that they cannot conduct good regularization by themselves. This is contrast to VAE, confirming the importance of the probabilistic codes: without this probabilistic nature, it would be impossible to model the complex distribution of x-vectors.

Table 1: *Performance (EER%) on SITW Dev. Core and Eval. Core.*

SITW Dev. Core

	Cosine	PCA	PLDA	L-PLDA	P-PLDA
x-vector	15.67	16.17	9.09	**3.12**	4.16
a-vector	16.10	16.48	11.21	4.24	5.01
v-vector	10.32	9.94	3.62	3.54	4.31
c-vector	**9.05**	**8.55**	**3.50**	3.31	**3.85**

SITW Eval. Core

	Cosine	PCA	PLDA	L-PLDA	P-PLDA
x-vector	16.79	17.22	9.16	3.80	4.84
a-vector	16.05	16.81	12.14	4.27	5.09
v-vector	10.11	10.03	**3.64**	3.64	4.43
c-vector	**9.05**	**8.83**	3.77	**3.53**	**4.10**

3.4. Analysis

To better understand the VAE-based regularization, we compute the skewness and kurtosis of the distributions of different speaker codes. The skewness and kurtosis are defined as follows:

$$\mathrm{Skew}(x) = \frac{E[(x-\mu_x)^3]}{\sigma_x^3}\ ,\ \mathrm{Kurt}(x) = \frac{E[x-\mu_x]^4}{\sigma_x^4} - 3,$$

where μ_x and σ_x denote the mean and standard variation of x, respectively. More Gaussian is a distribution, more close to zero are the two values.

The utterance-level and speaker-level skewness and kurtosis of different speaker codes are reported in Table 2. Focusing on the utterance-level results, it can be seen that the values of skewness and kurtosis of both v-vector and c-vector are clearly smaller than x-vector. This means that the v-vector and the c-vector are more Gaussian. For the speaker-level results, it can be found that the kurtosis was largely reduced in v-vectors and c-vectors. This indicates that the Gaussian regularization placed by VAE on the marginal has implicitly regularized the prior, which is the major reason that these vectors are more suitable for PLDA. The a-vector, derived from AE, has smaller skewness but larger kurtosis compared to the x-vector, on both the utterance-level and the speaker-level, suggesting that AE did not perform a good regularization.

Table 2: *Utterance-level and speaker-level skewness and kurtosis of different speaker codes on the Voxceleb set.*

	Skew(utt)	Kurt(utt)	Skew(spk)	Kurt(spk)
x-vector	-0.0423	-0.3604	0.0018	-0.4499
a-vector	-0.0072	-0.7740	0.0014	-0.9765
v-vector	-0.0055	0.1324	-0.0042	-0.0285
c-vector	-0.0043	0.1154	-0.0076	-0.0298

3.5. Out-of-domain test

In this experiment, we test the performance of various systems on the CSLT-SITW dataset. Due to the limited data, three-fold cross-validation was used whenever training is required. Three experiments were conducted: (1) directly using all the front-end and back-end models trained by VoxCeleb; (2) retraining all the models except the x-vector DNN; (3) the same as the retraining scheme, but all the PLDA models were trained by an unsupervised adaptation [23]. The results show that scheme (2) is generally the best, and the PLDA adaptation contributes additional gains in some test settings. For simplicity, only the retraining results under scheme (2) are reported in Table 3. The results exhibit a similar trend as in the SITW test, that both the v-vector and c-vector outperform the x-vector, and the c-vector obtained the best performance in nearly all the test settings. Compared to the SITW test, the larger performance gains obtained by VAE-regularization. It might be attributed to the more complex acoustic conditions of CSLT-SITW, though more investigation is required.

Table 3: *Performance (EER%) on CSLT-SITW.*

	Cosine	PCA	PLDA	L-PLDA	P-PLDA
x-vector	16.65	16.89	16.91	15.39	13.29
v-vector	13.55	13.71	**12.46**	12.06	12.02
c-vector	**12.98**	**13.13**	12.48	**12.01**	**11.98**

4. Conclusions

This paper proposed a VAE-based regularization for deep speaker embedding. By this model, x-vectors that usually exhibit a complex distribution are mapped to latent speaker codes that are simply Gaussian. This model was further enhanced by a speaker cohesive loss, which regularizes speaker conditionals. Experiments on the SITW dataset and a private commercial dataset demonstrated that the VAE-regularized speaker codes can achieve better performance with either cosine distance or PLDA scoring, compared to the x-vector baseline. Future work will investigate speaker-aware VAE, where speaker codes and utterance codes are hierarchically linked as in PLDA.

5. References

[1] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital signal processing*, vol. 10, no. 1-3, pp. 19–41, 2000.

[2] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

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

[4] P. Kenny, "Joint factor analysis of speaker and session variability: Theory and algorithms," Tech. Rep., 2005.

[5] S. Ioffe, "Probabilistic linear discriminant analysis," pp. 531–542, 2006.

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

[7] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Twelfth Annual Conference of the International Speech Communication Association*, 2011.

[8] E. Variani, X. Lei, E. McDermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2014, pp. 4052–4056.

[9] G. Heigold, I. Moreno, S. Bengio, and N. Shazeer, "End-to-end text-dependent speaker verification," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2016, pp. 5115–5119.

[10] L. Li, Y. Chen, Y. Shi, Z. Tang, and D. Wang, "Deep speaker feature learning for text-independent speaker verification," in *Interspeech*, 2017, pp. 1542–1546.

[11] D. Snyder, D. Garcia-Romero, G. Sell, D. Povey, and S. Khudanpur, "X-vectors: Robust dnn embeddings for speaker recognition," in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 5329–5333.

[12] S.-X. Zhang, Z. Chen, Y. Zhao, J. Li, and Y. Gong, "End-to-end attention based text-dependent speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 171–178.

[13] D. Snyder, P. Ghahremani, D. Povey, D. Garcia-Romero, Y. Carmiel, and S. Khudanpur, "Deep neural network-based speaker embeddings for end-to-end speaker verification," in *2016 IEEE Spoken Language Technology Workshop (SLT)*. IEEE, 2016, pp. 165–170.

[14] D. Snyder, D. Garcia-Romero, A. McCree, G. Sell, D. Povey, and S. Khudanpur, "Spoken language recognition using x-vectors," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 105–111. [Online]. Available: http://dx.doi.org/10.21437/Odyssey.2018-15

[15] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," in *Proc. Odyssey 2018 The Speaker and Language Recognition Workshop*, 2018, pp. 74–81. [Online]. Available: http://dx.doi.org/10.21437/Odyssey.2018-11

[16] L. Li, Z. Tang, Y. Shi, and D. Wang, "Gaussian-constrained training for speaker verification," in *2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[18] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *arXiv preprint arXiv:1706.08612*, 2017.

[19] D. Snyder, G. Chen, and D. Povey, "MUSAN: A Music, Speech, and Noise Corpus," 2015.

[20] T. Ko, V. Peddinti, D. Povey, M. L. Seltzer, and S. Khudanpur, "A study on data augmentation of reverberant speech for robust speech recognition," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 5220–5224.

[21] M. McLaren, L. Ferrer, D. Castan, and A. Lawson, "The speakers in the wild (SITW) speaker recognition database." in *Interspeech*, 2016, pp. 818–822.

[22] D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz *et al.*, "The kaldi speech recognition toolkit," in *IEEE 2011 workshop on automatic speech recognition and understanding*, no. EPFL-CONF-192584. IEEE Signal Processing Society, 2011.

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