INTERSPEECH 2021
30 August – 3 September, 2021, Brno, Czechia

Gradient Regularization for Noise-Robust Speaker Verification

Jianchen Li, Jiqing Han, Hongwei Song

School of Computer Science and Technology, Harbin Institute of Technology, Harbin, China
{lijianchen, jqhan, songhongwei}@hit.edu.cn

Abstract

Noise robustness is a challenge for speaker recognition systems. To solve this problem, one of the most common approaches is to joint-train a model by using both clean and noisy utterances. However, the gradients calculated on noisy utterances generally contain speaker-irrelevant noisy components, resulting in overfitting for the seen noisy data and poor generalization for the unseen noisy environments. To alleviate this problem, we propose the gradient regularization method to reduce the speaker-irrelevant noisy components by aligning the gradients among the noisy utterances and their clean counterparts. Specifically, the gradients on noisy utterances are forced to follow the directions of the gradients calculated on their clean counterparts, and the gradients across different types of noisy utterances are also aligned to point in similar directions. Since the noise-related components of the gradients can be reduced by the above alignment, the speaker model can be prevented from encoding irrelevant noisy information. To achieve the gradient regularization goals, a novel sequential inner training strategy is also proposed. Experiments on the VoxCeleb1 dataset indicate that our method achieves the best performance in seen and unseen noisy environments.

Index Terms: gradient regularization, noise-robust, speaker verification, speaker recognition

1. Introduction

Automatic Speaker Verification (ASV) is the task of verifying if an utterance is spoken by a claimed speaker [1]. Recently, with the development of deep learning, the state-of-the-art ASV system has shifted from i-vector [2] with probabilistic linear discriminant analysis (PLDA) [3] to deep speaker embedding models [4, 5]. Although the deep speaker embedding model has achieved significant success in the clean acoustic environment [6], the performance tends to degrade when the model is deployed in realistic complex environments, especially noisy environments. In general, there is no prior knowledge of noise distribution in advance, which makes the noise problem more difficult [7].

Tremendous research efforts have been invested to improve noise robustness. One approach is to apply speech enhancement techniques to recover the clean signals. Zhao *et al.* [8] first proposed deep learning based speech enhancement modules for speaker recognition. Shon *et al.* [9] proposed the VoiceID loss for generating ratio masks to filter out the unnecessary components of the spectrogram. Shi *et al.* [10] integrated speech enhancement and speaker recognition modules into one framework. Another approach regards noisy data as a different domain from clean data and uses adversarial training to get domain-invariant speaker models, in which the domain label of training data can be various noise types or different SNRs [11]. For example, the multitask adversarial training framework was proposed for training noise-robust speaker models [12], and the unsupervised adversarial invariance architecture was adopted to disentangle speaker-discriminative information [13]. There are also other robust approaches. Kataria *et al.* [14] optimized a deep feature loss for feature-domain denoising. Kim *et al.* [15] proposed the orthogonal vector pooling strategy to remove irrelevant factors.

In addition to the above approaches, one of the most commonly used approach is joint training, which trains a single model on the mixed dataset consisting of clean utterances and noisy utterances obtained by data augmentation [16, 4]. In most cases, joint training can achieve satisfactory results [12]. However, the speaker-irrelevant noisy information is usually encoded by the network during training process [17], causing the model to overfit on the seen noisy utterances. Thus poor generalization may be observed when facing an unseen noisy environment.

In this paper, we propose the gradient regularization method to prevent the network from encoding speaker-irrelevant noisy information. Our method is based on the intuitive idea that the gradient vectors calculated on a clean utterance and its noisy counterparts should be in the same direction since these utterances contain the same acoustic content for recognizing speakers. Specifically, the gradient regularization terms are appended to the joint training loss to maximize the similarity of the gradient vectors on clean utterances and their noisy counterparts. Meanwhile, the similarity of the gradient vectors across different types of noisy utterances is also maximized to suppress the noise-related components of the gradients. In this manner, the optimizer will find an optimization route where the gradients on clean utterances and their noisy counterparts are in the similar directions at all points along the route, thus the noise-related components of the gradients on noisy utterances can be largely reduced, ensuring that the main direction of the gradient is relevant to learning speaker-discriminative embeddings. Furthermore, a new sequential inner training strategy is also proposed to achieve the optimization goal.

2. Related Works

2.1. Deep Speaker Embedding Models

In general, the deep speaker embedding framework consists of a frame-level feature extractor, an utterance-level encoding layer and several fully connected layers for dimensionality reduction. The most commonly used feature extraction network is time-delayed neural network (TDNN) [18] or convolutional neural network (CNN) [19], it maps variable-length input speech frames to intermediate frame-level deep hidden features. Then the encoding layer (e.g. average pooling layer [20], statistic pooling layer [6]) aggregates all frame-level features into a low-dimensional utterance-level feature. The fully connected layers and the loss function are employed to train the whole network. Our method is based on the original speaker embedding model, and improves the noise robustness by introducing the regular-

http://dx.doi.org/10.21437/Interspeech.2021-1216

ization term into the loss function.

2.2. MLDG-based Method

Our method is related to the recently proposed Meta-Learning Domain Generalization (MLDG) method [21], which is a meta-learning based approach for training the domain-invariant models in computer vision. Inspired by MLDG, we propose the gradient regularization method to achieve noise robustness. We will elaborate on their differences and highlight the advantages of our approach later in section 3.1.

3. Proposed Methods

Suppose we have the clean dataset and K noisy version datasets, where K is the number of noise types. At each learning iteration, we sample clean batch $\mathcal{D}_0$ and its K noisy counterparts $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_K$. The empirical risk on each batch can be written as $\mathcal{L}(\theta; \mathcal{D}_j) = \frac{1}{B}\sum_{i=1}^{B} \ell\left(f_\theta\left(\mathbf{x}_i^j\right), y_i^j\right)$, where $j = 0, 1, \ldots, K$, f_θ is the embedding network with parameters θ, y is the speaker label, B is the batch size and $\ell(\cdot)$ is the cross-entropy loss with softmax function. The joint training loss aggregates the loss on each batch. And the gradient on each batch can be expressed as $g_j = \partial\mathcal{L}(\theta; \mathcal{D}_j)/\partial\theta$. Ideally, these gradient vectors should be in the similar directions. However, as show in Figure 1(a), the gradient vector on the noisy batch g_k or g_l has the component of encoding the speaker-irrelevant noisy information.

3.1. Gradient Regularization

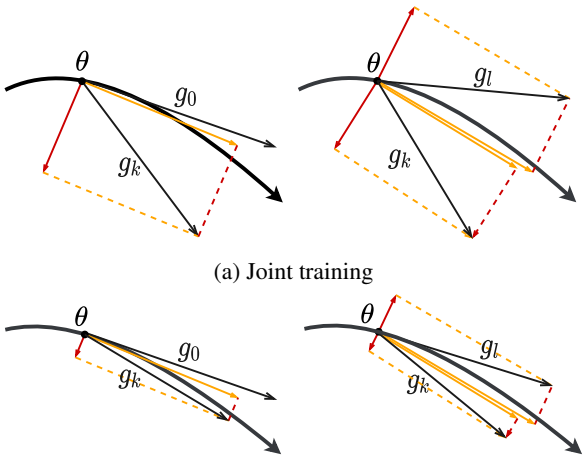
To suppress the noise-related components of the gradients on noisy batches, the proposed gradient regularization loss is composed of two terms. One is to force the gradients on the noisy batches to follow the same direction as the gradient calculated on clean counterpart, the other is to align all pairwise combinations of gradients on K noisy batches. The gradient regularization loss can be formulated as

$$\mathcal{L}_{GR} = -\lambda_1 \sum_{k=1}^{K} g_0 \cdot g_k - \lambda_2 \sum_{k=2}^{K}\sum_{l=1}^{k-1} g_l \cdot g_k \qquad (1)$$

where g_0 is the gradient on the clean batch, g_k or g_l is the gradient on the noisy batch, λ_1 and λ_2 are trade-off parameters.

The first term in (1) maximize the inner product of g_0 and its noisy counterparts g_k. Since the gradient on the clean batch is more reliable than its noisy counterparts and the main direction of g_0 is relevant to learning speaker representations, aligning g_k to g_0 can reduce the noise-related component of g_k. Note that g_0 is treated as a constant vector in each update step. More concretely, the gradient of each inner product term $g_0 \cdot g_k$ can be expressed as $H_0 g_k + H_k g_0$, where H_k is the Hessian matrix of $\mathcal{L}(\theta; \mathcal{D}_k)$. Intuitively, it can be understood as pushing the gradient vector towards each other according to the second-order derivative information. Since our purpose is to reduce the noise-related component of g_k, we omit the term $H_0 g_k$ to prevent the noisy information of g_k from disturbing the learning direction on the clean batch.

The second term in (1) maximize the similarity of the gradients across different types of noisy batches. In reality, it is very expensive to collect completely clean utterances. Thus, when the utterances used to calculate g_0 are not strictly in clean conditions, we align all gradient pairs on the noisy batches to further reduce the noise-related components. The same speaker-related acoustic content of these different types of noisy utterances ensures that the aligned gradient direction is mainly related to learning speaker representations.



(a) Joint training

(b) Joint training with L_{GR}

Figure 1: *Optimization routes of (a) joint training and (b) joint training with gradient regularization. g_0 is the gradients on clean batch, g_k, g_l are the gradients on noisy batches. The orange solid line represents the component of encoding the speaker information, and the red solid line represents the component of encoding the noisy information.*

Finally, appending $\mathcal{L}_{GR}$ to the joint training loss, the total loss function can be written as

$$\mathcal{L} = \sum_{k=0}^{K} \mathcal{L}(\theta; \mathcal{D}_k) + \mathcal{L}_{GR} \qquad (2)$$

From Figure 1(b), we can see that this loss can largely reduce the noise-related component of the gradient on the noisy batch.

At this point, we would like to illustrate the difference between the MLDG [21] and our method. The MLDG is to find an optimization route with aligned gradients by maximizing the similarity of the gradients between one domain and remaining source domains. Compared with our method, g_0 is not treated as a constant vector in the MLDG method to prevent the noisy information of g_k from disturbing the direction of g_0, and all pairs of g_k and g_l are not aligned to further reduce the noise-related components.

3.2. Sequential Inner Training (SIT) Strategy

The optimization of (2) requires calculating the second-order gradients of $\mathcal{L}$, which is hard to directly compute. To avoid this, a novel sequential inner training strategy is proposed. For easy understanding, we first present the SIT algorithm, then we show that it appropriately optimizes (2) in this section.

The SIT strategy involves an inner loop to produce the higher-order gradients of $\mathcal{L}$ and an outer loop to update the parameters θ. At each iteration, assume the initial point is $\overline{\theta}_0$, and the gradient w.r.t $\overline{\theta}_0$ is $g_k = \partial\mathcal{L}\left(\overline{\theta}_0; \mathcal{D}_k\right)/\partial\theta$. In the inner loop, $\overline{\theta}_0$ is first updated on the clean batch $\overline{\theta}_1 = \overline{\theta}_0 - \lambda_1 g_0$, and then sequentially updated on the noisy batches in random order $\overline{\theta}_{k+1} = \overline{\theta}_k - 2\lambda_2 \overline{g}_k$, where $\overline{g}_k = \partial\mathcal{L}\left(\overline{\theta}_k; \mathcal{D}_k\right)/\partial\theta$. When the final trained parameter $\overline{\theta}_{K+1}$ is obtained, the offset vector

Table 1: *EER(%) and DCF of various systems under the seen noisy environments. Best in bold.*

Seen Noise Types	SNR (dB)	Clean Training		Joint Training		VI Loss [17]		MLDG [21]		Our Method	
		EER(%)	DCF	EER(%)	DCF	EER(%)	DCF	EER(%)	DCF	EER(%)	DCF
Clean	-	5.32	0.536	5.07	0.494	4.77	**0.459**	4.94	0.501	**4.57**	0.475
Ambient Noise	0	16.85	0.945	10.42	0.763	9.57	0.753	10.04	**0.740**	**9.49**	0.754
	5	12.14	0.834	8.22	0.670	7.35	0.682	7.92	0.672	**7.31**	**0.658**
	10	9.15	0.740	6.82	0.632	6.21	0.610	6.69	0.637	**6.12**	**0.601**
	15	7.33	0.635	6.04	0.593	5.51	0.554	5.98	0.588	**5.48**	**0.546**
	20	6.28	0.577	5.66	0.557	5.23	**0.504**	5.49	0.548	**5.04**	0.526
Music	0	21.92	0.988	13.16	0.865	12.06	0.844	12.60	0.820	**11.82**	**0.817**
	5	13.22	0.882	8.41	0.715	7.83	0.699	8.27	0.721	**7.71**	**0.686**
	10	8.48	0.670	6.50	0.608	6.03	0.614	6.36	0.640	**5.70**	**0.594**
	15	6.35	0.580	5.60	0.541	5.17	0.549	5.50	0.554	**5.00**	**0.520**
	20	5.64	0.534	5.28	0.527	4.90	0.515	5.14	0.533	**4.76**	**0.476**
Babble	0	43.21	0.999	37.73	0.990	36.80	0.989	**36.46**	**0.988**	36.79	0.997
	5	31.75	0.992	21.00	0.937	20.25	**0.921**	19.95	0.926	**19.27**	0.934
	10	18.16	0.934	10.99	0.755	10.30	0.778	10.30	0.785	**9.96**	**0.750**
	15	9.94	0.795	7.60	0.645	7.05	0.632	7.32	0.647	**6.74**	**0.618**
	20	6.99	0.648	6.08	0.559	5.55	0.558	5.98	0.563	**5.44**	**0.530**
All Noises	-	15.30	0.978	10.87	0.729	10.21	0.710	10.55	0.721	**9.93**	**0.709**

$\frac{1}{\lambda_1}\left(\theta-\overline{\theta}_1\right)+\frac{1}{2\lambda_2}\left(\overline{\theta}_1-\overline{\theta}_{K+1}\right)$ is used as the gradient for updating θ in the outer loop. The details of the strategy are shown in Algorithm 1.

We then show that Algorithm 1 can optimize (2). After k updates, $\overline{\theta}_k$ can be written as $\overline{\theta}_k=\overline{\theta}_0-\lambda_1 g_0-2\lambda_2\sum_{l=1}^{k-1}\overline{g}_l$. Thus the Taylor series of $\overline{g}_k$ at $\overline{\theta}_0$ is

$$\begin{aligned}\overline{g}_k &= g_k+H_k\left(\overline{\theta}_k-\overline{\theta}_0\right)+O\left(\left\|\overline{\theta}_k-\overline{\theta}_0\right\|^2\right)\\ &= g_k-\lambda_1 H_k g_0-2\lambda_2\sum_{l=1}^{k-1}H_k g_l+O\left(\lambda^2\right)\end{aligned} \tag{3}$$

where the last step is obtained by $\overline{g}_l=g_l+O(\lambda)$, H_k is the Hessian matrix of $\mathcal{L}\left(\overline{\theta}_0;\mathcal{D}_k\right)$, $\lambda=\lambda_1+2\lambda_2$. Bring $\overline{g}_k$ into the offset vector expression, we can get

$$\begin{aligned}&\frac{1}{\lambda_1}\left(\theta-\overline{\theta}_1\right)+\frac{1}{2\lambda_2}\left(\overline{\theta}_1-\overline{\theta}_{K+1}\right)=g_0+\sum_{k=1}^{K}\overline{g}_k\\ &=\sum_{k=0}^{K}g_k-\lambda_1\sum_{k=1}^{K}H_k g_0-2\lambda_2\sum_{k=2}^{K}\sum_{l=1}^{k-1}H_k g_l+O(\lambda^2)\end{aligned} \tag{4}$$

According to [22], if we randomize the order of noisy batches at each iteration, $\mathbb{E}\left[H_k g_l\right]=\mathbb{E}\left[H_l g_k\right]=\frac{1}{2}\frac{\partial(g_l\cdot g_k)}{\partial\theta}$. Therefore, this training strategy can appropriately optimize (2).

Algorithm 1 Sequential Inner Training Strategy

Input: clean dataset and K noisy version datasets
Init: embedding network parameters θ, hyperparameters α, λ

while not done **do** # outer loop
  $\overline{\theta}_0=\theta$
  Sample batches $\mathcal{D}_0$, $[\mathcal{D}_1,\mathcal{D}_2,\ldots,\mathcal{D}_K]$
  Shuffle $([\mathcal{D}_1,\mathcal{D}_2,\ldots,\mathcal{D}_K])$
  $\overline{\theta}_1=\overline{\theta}_0-\lambda_1 g_0$
  for k in $[1,2,\ldots,K]$ **do** # inner loop
    $\overline{\theta}_{k+1}=\overline{\theta}_k-2\lambda_2\overline{g}_k$
  end for
  $\theta\leftarrow\theta-\frac{\alpha}{\lambda_1}\left(\theta-\overline{\theta}_1\right)-\frac{\alpha}{2\lambda_2}\left(\overline{\theta}_1-\overline{\theta}_{K+1}\right)$
end while

4. Experiments

4.1. Datasets

Following the common experiment settings [9, 17, 10], we conduct the experiments on the VoxCeleb1 [20] dataset. The training set contains 148,642 training utterances from 1211 speakers. The test set contains 4,874 utterances from 40 speakers, which constructs 37720 test trials. Although this dataset is not strictly in clean conditions, we assume it as clean corpus and generate the related noisy utterances [17].

A noise-corrupted version of the VoxCeleb1 dataset is generated by artificially adding different types of noise data. The MUSAN [23] dataset is used as noise source, which contains 60 hours of speech, 42 hours of music and 6 hours ambient noise. This noise dataset is divided into two non-overlapping subsets for generating noisy training and testing utterances respectively. At the training stage, for each clean utterance, three noisy utterances (ambient noise, music and babble) are generated at the random SNR level from 0dB to 20dB. This forms a mixed training set with a 3:1 ratio of noisy utterances to clean utterances. The babble noise is constructed by mixing three to six speech files into one. At the testing stage, we evaluate the performance of the speaker verification systems under seen and unseen noisy environments. For the seen noisy environments, the noise data (ambient noise, music and babble) are sampled from the remaining half of the MUSAN dataset. For the unseen noisy environments, we select *cafeteria* and *train station* noises from the BBC Noise dataset[1] as another noise source to generate noisy testing utterances. We also combine all noisy testing trials to form "All Noises" trials.

[1] `http://bbcsfx.acropolis.org.uk/`

Table 2: *EER(%) and DCF of various systems under the unseen noisy environments. Best in bold.*

Unseen Noise Types	SNR (dB)	Clean Training		Joint Training		VI Loss [17]		MLDG [21]		Our Method	
		EER(%)	DCF	EER(%)	DCF	EER(%)	DCF	EER(%)	DCF	EER(%)	DCF
Cafeteria	0	17.39	0.949	12.01	0.824	11.24	0.848	11.89	0.834	**11.09**	**0.811**
	5	10.60	0.834	8.03	0.718	7.43	0.688	8.10	0.687	**7.42**	**0.667**
	10	7.41	0.643	6.21	0.615	5.86	0.598	6.24	**0.576**	**5.80**	0.586
	15	6.08	0.586	5.46	0.568	5.18	0.514	5.37	**0.510**	**5.05**	0.520
	20	5.57	0.545	5.19	0.536	4.85	**0.457**	5.02	0.506	**4.82**	0.484
Train Station	0	18.67	0.963	11.36	0.814	10.77	**0.803**	11.41	0.810	**10.65**	0.809
	5	11.26	0.806	8.21	0.664	7.55	0.675	8.11	**0.648**	**7.36**	0.657
	10	7.60	0.621	6.54	0.581	6.06	0.568	6.41	0.563	**5.81**	**0.554**
	15	6.29	0.564	5.73	0.532	5.31	0.501	5.56	0.519	**5.03**	**0.499**
	20	5.65	0.549	5.31	0.496	4.94	0.481	5.20	0.520	**4.74**	**0.479**
All Noises	-	10.84	0.924	7.61	0.697	7.00	0.674	7.42	0.682	**6.79**	**0.652**

4.2. Implementation Details

For input features, 23-dimensional MFCC and 3-dimensional pitch features are extracted within a 25ms sliding window with a hop size of 10ms. Cepstral mean normalization (CMN) is performed within a 3 second sliding window and energy-based voice activity detector (VAD) is used to remove silence frames. Each utterance is cut into 200-frame chunks with 10% overlap.

TDNN [18] network is used as the speaker embedding extractor for its simplicity. However, our method is agnostic to backbone networks. After training, the 128-dimensional speaker embeddings are extracted from the penultimate layer of the network, and the cosine similarity is used for scoring. The equal error rate (EER) [24] and the detection cost function (DCF) [24] with $P_{\text{target}} = 0.01$ is used as the performance metric. The reported DCF is adopted in NIST SRE 2018 [25] and VoxSRC 2019 [26].

For optimization, AdamW [27] optimizer with the learning rate of 0.001 and the weight decay of 0.3 is used to train the whole network. ReduceLROnPlateau scheduler in Pytorch [28] is adopted to update the learning rate. The trade-off parameters λ_1 and λ_2 in (1) are initialized to 1e-3 and 5e-4 respectively, then updated in proportion to the learning rate. The batch size B is 64 in this work. For final test, we use the best performing model on the validation set.

4.3. Results

Table 1 and Table 2 show the performance under the seen and unseen noisy conditions, respectively. Where clean training means the model is trained on the original dataset and joint training (Baseline) means the model is trained on the mixed dataset. The model trained with the variability-invariant (VI) loss [17] is used to compare. The VI loss minimizes the distance of embeddings between clean and noisy utterances, which is another recently proposed improvement approach for joint training loss. We re-implement it under the fair experimental setting. We also compare the model that directly apply the MLDG method to improve noise robustness.

Table 1 illustrates that our method can achieve the best results under the clean and seen noisy conditions. And the MLDG method have only limited improvements compared to the joint training under the clean and low SNR level conditions, since the gradient information on the noisy batch will disturb the learning direction on its clean counterpart.

Table 3: *Ablation study of the proposed method*

	The 1st term	The 2nd term	EER(%)	DCF
Baseline	-	-	7.61	0.697
Our	✓	-	6.81	0.678
	-	✓	6.99	0.669
	✓	✓	**6.79**	**0.652**

Table 2 shows that our proposed method outperforms all other models under the unseen noisy environments. The VI loss only aligns the distribution of embeddings between clean and noisy utterances. Compared with this method, our method not only aligns the gradients between clean and noisy utterances, but also aligns the gradient among different types of noisy utterances, thus achieving better performance. Compared with the baseline, our method achieves 10.8% and 6.5% reduction in terms of EER and DCF respectively. It indicates that our method can avoid learning the speaker-irrelevant noisy information from seen noisy environments and improve the generalization ability in unseen noisy environments.

Ablation study is conducted to show the influences of the first and second terms in $\mathcal{L}_{GR}$ on the performance. The results of "All Noises" trials under the unseen noisy environments are shown in Table 3. This shows that adding either term boosts the performance and combining the two gives best performance.

5. Conclusions

In this work, we proposed two gradient regularization terms to prevent the network from encoding speaker-irrelevant noisy information. A novel sequential inner training strategy was also proposed to achieve the optimization goal. Experimental results indicated that our method can avoid learning the speaker-irrelevant noisy information from seen noisy environments, and achieve the best generalization performance in unseen noisy environments.

6. Acknowledgements

This research is supported by the National Natural Science Foundation of China under grant No. U1736210.

7. References

[1] J. H. Hansen and T. Hasan, "Speaker recognition by machines and humans: A tutorial review," *IEEE Signal processing magazine*, vol. 32, no. 6, pp. 74–99, 2015.

[2] 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, 2010.

[3] S. Ioffe, "Probabilistic linear discriminant analysis," in *European Conference on Computer Vision*. Springer, 2006, pp. 531–542.

[4] 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.

[5] C. Li, X. Ma, B. Jiang, X. Li, X. Zhang, X. Liu, Y. Cao, A. Kannan, and Z. Zhu, "Deep speaker: an end-to-end neural speaker embedding system," *arXiv preprint arXiv:1705.02304*, vol. 650, 2017.

[6] 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.

[7] T. F. Zheng and L. Li, *Robustness-related issues in speaker recognition*. Springer, 2017.

[8] X. Zhao, Y. Wang, and D. Wang, "Robust speaker identification in noisy and reverberant conditions," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 22, no. 4, pp. 836–845, 2014.

[9] S. Shon, H. Tang, and J. Glass, "Voiceid loss: Speech enhancement for speaker verification," *arXiv preprint arXiv:1904.03601*, 2019.

[10] Y. Shi, Q. Huang, and T. Hain, "Robust speaker recognition using speech enhancement and attention model," *arXiv preprint arXiv:2001.05031*, 2020.

[11] Z. Meng, Y. Zhao, J. Li, and Y. Gong, "Adversarial speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6216–6220.

[12] J. Zhou, T. Jiang, L. Li, Q. Hong, Z. Wang, and B. Xia, "Training multi-task adversarial network for extracting noise-robust speaker embedding," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6196–6200.

[13] R. Peri, M. Pal, A. Jati, K. Somandepalli, and S. Narayanan, "Robust speaker recognition using unsupervised adversarial invariance," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6614–6618.

[14] S. Kataria, P. S. Nidadavolu, J. Villalba, N. Chen, P. Garcia-Perera, and N. Dehak, "Feature enhancement with deep feature losses for speaker verification," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 7584–7588.

[15] I. Kim, K. Kim, J. Kim, and C. Choi, "Deep speaker representation using orthogonal decomposition and recombination for speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6126–6130.

[16] Y. Lei, L. Burget, and N. Scheffer, "A noise robust i-vector extractor using vector taylor series for speaker recognition," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*. IEEE, 2013, pp. 6788–6791.

[17] D. Cai, W. Cai, and M. Li, "Within-sample variability-invariant loss for robust speaker recognition under noisy environments," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 6469–6473.

[18] D. Snyder, D. Garcia-Romero, D. Povey, and S. Khudanpur, "Deep neural network embeddings for text-independent speaker verification." in *Interspeech*, 2017, pp. 999–1003.

[19] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *Proc. Interspeech 2018*, pp. 1086–1090, 2018.

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

[21] D. Li, Y. Yang, Y.-Z. Song, and T. Hospedales, "Learning to generalize: Meta-learning for domain generalization," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[22] A. Nichol, J. Achiam, and J. Schulman, "On first-order meta-learning algorithms," *arXiv preprint arXiv:1803.02999*, 2018.

[23] D. Snyder, G. Chen, and D. Povey, "Musan: A music, speech, and noise corpus," *arXiv preprint arXiv:1510.08484*, 2015.

[24] D. Snyder, D. Garcia-Romero, G. Sell, A. McCree, D. Povey, and S. Khudanpur, "Speaker recognition for multi-speaker conversations using x-vectors," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5796–5800.

[25] S. O. Sadjadi, C. Greenberg, E. Singer, D. Reynolds, L. Mason, and J. Hernandez-Cordero, "The 2018 NIST Speaker Recognition Evaluation," in *Proc. Interspeech 2019*, 2019, pp. 1483–1487. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2019-1351

[26] J. S. Chung, A. Nagrani, E. Coto, W. Xie, M. McLaren, D. A. Reynolds, and A. Zisserman, "Voxsrc 2019: The first voxceleb speaker recognition challenge," *arXiv preprint arXiv:1912.02522*, 2019.

[27] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[28] 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*, 2019, pp. 8026–8037.