*INTERSPEECH 2020*
October 25–29, 2020, Shanghai, China

# Dynamic Margin Softmax Loss for Speaker Verification

*Dao Zhou[1], Longbiao Wang[1,*], Kong Aik Lee[2], Yibo Wu[1], Meng Liu[1], Jianwu Dang[1,3,*], Jianguo Wei[1]*

[1]Tianjin Key Laboratory of Cognitive Computing and Application,
College of Intelligence and Computing, Tianjin University, Tianjin, China
[2]Biometrics Research Laboratories, NEC Corporation, Japan
[3]Japan Advanced Institute of Science and Technology, Ishikawa, Japan
{zhoudao, longbiao_wang}@tju.edu.cn, jdang@jaist.ac.jp

## Abstract

We propose a dynamic-margin softmax loss for the training of deep speaker embedding neural network. Our proposal is inspired by the additive-margin softmax (AM-Softmax) loss reported earlier. In AM-Softmax loss, a constant margin is used for all training samples. However, the angle between the feature vector and the ground-truth class center is rarely the same for all samples. Furthermore, the angle also changes during training. Thus, it is more reasonable to set a dynamic margin for each training sample. In this paper, we propose to dynamically set the margin of each training sample commensurate with the cosine angle of that sample, hence, the name dynamic-additive-margin softmax (DAM-Softmax) loss. More specifically, the smaller the cosine angle is, the larger the margin between the training sample and the corresponding class in the feature space should be to promote intra-class compactness. Experimental results show that the proposed DAM-Softmax loss achieves state-of-the-art performance on the VoxCeleb dataset by 1.94% in equal error rate (EER). In addition, our method also outperforms AM-Softmax loss when evaluated on the Speakers in the Wild (SITW) corpus.

**Index Terms**: speaker verification, large-margin loss, intra-class compactness

## 1. Introduction

Automatic speaker verification (ASV) becomes increasingly popular for biometric authentication due to its convenience and effectiveness. The aim of an ASV system is to authenticate the identity of a speaker given his/her utterances. The ASV task encompasses both text-dependent and text-independent modes depending whether or not the content of utterances is constrained. The ASV pipeline consisting of a speaker embedding [1, 2, 3, 4] front-end followed by a Probabilistic Linear Discriminant Analysis [5] back-end has been dominant over the past years.

Recently, using deep neural networks (DNNs) to extract discriminative speaker embeddings has attracted much attention. Compared to i-vector [1], deep-learning based embeddings have shown superior performance on a wide variety of ASV tasks [2, 3, 4]. In this regard, most previous works have focused on searching for network architectures that produce speaker embedding vectors with improved representation power. In [6], a long-short-term-memory (LSTM) layer was incorporated into the x-vector's time delay neural network [4] to extract more comprehensive speaker information. In [7], frame-level features were aggregated into utterance-level embeddings by incorporating NetVLAD and GhostVLAD layers into the 'thin-ResNet' architecture, which achieved better performance compared with the standard ResNet [8]. More recently, generative adversarial networks were also successfully applied to deal with the problems of short utterances and domain mismatch [9, 10].

As in most machine learning tasks, a good loss function for speaker embedding would enlarge inter-class variations while intra-class variation is reduced. In contrary, the learned embeddings from the softmax loss are optimized for inter-class separation alone without taking into account intra-class compactness. A number of novel loss functions were proposed to address this issue, for example, triplet loss and some variants of softmax loss. Triplet loss [11, 12] optimizes the embedding space by minimizing the distance between the feature pairs from the same speaker at the same time it also minimizes the distance between the feature pairs from different speakers. The downside is that triplet pairs mining is by itself a difficult problem. As a variant of softmax loss, angular softmax loss [13] has been shown to perform well on the ASV task [14]. More recently, the cosine margin was introduced in [15], which performs better than other loss functions [16, 17, 18].

In [15], the cosine margin is manually tuned and applied to all training samples. This is suboptimal since the angle between the feature vector and the center of the ground-truth class is hardly the same for each individual sample. Furthermore, the angle also changes during training. Thus, it is more reasonable to set a dynamic margin for each training sample. In this paper, we propose a dynamic cosine margin softmax. In the proposed method, the margin of a training sample is negatively correlated with the cosine angle of that sample. More specifically, the smaller the cosine angle is, the larger the margin between the training sample and the corresponding class in the feature space, whcih encourages better intra-class compactness. Experiments on the VoxCeleb and SITW datasets [3, 8, 19] indicate the efficacy of the proposed DAM-Softmax loss, the relative error reduction is between 4.9%-8.1% for these datasets compared with AM-Softmax loss.

The rest of this paper is organized as follows. Section 2 reviews the conventional softmax loss, angular softmax loss and its variant. Section 3 introduces the proposed dynamic cosine angular loss. Experimental setup and results are presented in Section 4. Section 5 concludes the paper.

## 2. From softmax to angular softmax

### 2.1. Softmax loss

We start with the definition of the softmax loss in its basic form:

$$L_S = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{\mathbf{W}_{y_i}^T \mathbf{x}_i + b_{y_i}}}{\sum_{j=1}^{C} e^{\mathbf{W}_j^T \mathbf{x}_i + b_j}} \quad (1)$$

* Corresponding authors



http://dx.doi.org/10.21437/Interspeech.2020-1106

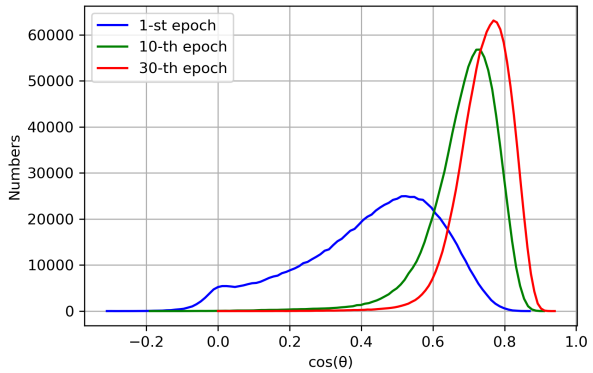


Figure 1: *The cosine angle $cos(\theta)$ of the training samples at three different training stages plotted as distribution.*

where $N$ is the number of training samples, $C$ is the number of classes, $\mathbf{x}_i$ denotes the feature representation of the $i$-th sample, and $y_i$ indicates the target class of this $i$-th sample. The quantity $\mathbf{W}_j$ denotes the weight vector of class $j$ while $b_j$ is the corresponding bias term. Using the basic rules of trigonometry, the expression $\mathbf{W}_{y_i}^T \mathbf{x}_i + b_{y_i}$ in the numerator on the right-hand-side of Eq. (1) can be rewritten as

$$\|\mathbf{W}_{y_i}\| \, \|\mathbf{x}_i\| \, cos(\theta_{y_i}) + b_{y_i} \tag{2}$$

in terms of the angle $\theta_{y_i}$ between the two vectors $\mathbf{W}_{y_i}$ and $\mathbf{x}_i$.

### 2.2. Angular softmax loss

From Eq. (2), we normalize the weight vector to have the unit norm and discard the bias term by setting $\|\mathbf{W}_{y_i}\| = 1$ and $b_{y_i} = 0$. This leads to the so-called angular softmax (A-Softmax) loss [13] defined as follows:

$$L_A = -\frac{1}{N} \sum_{i=1}^{N} log \frac{e^{\|\mathbf{x}_i\| \phi(\theta_{y_i})}}{e^{\|\mathbf{x}_i\| \phi(\theta_{y_i})} + \sum_{j=1; j \neq y_i}^{C} e^{\|\mathbf{x}_i\| cos(\theta_j)}} \tag{3}$$

To arrive at the above equation, the cosine angle is also replaced with a more elaborate function:

$$\phi(\theta) = (-1)^k cos(m\theta) - 2k \tag{4}$$

where $k \in [0, m-1]$ and $\theta \in [\frac{k\pi}{m}, \frac{(k+1)\pi}{m}]$. The parameter $m$ is a positive integer that controls the size of the angular margin in Eq. (3), thereby enforcing intra-class compactness. In [20], the authors showed that the A-Softmax loss is better at producing a more discriminative speaker embedding than the plain vanilla softmax loss.

### 2.3. Additive-margin softmax loss

The additive-margin softmax (AM-softmax) loss was further extended in [15] at two fronts. Firstly, the angular margin is imposed with an additive term $m$ instead of a multiplicative term:

$$\phi(\theta_{y_i}) = cos(\theta_{y_i}) - m \tag{5}$$

Secondly, the norm of the feature vectors $\|\mathbf{x}_i\|$ was replaced with a hyper-parameter $s$, while $\mathbf{x}_i$ is normalized to the unit norm. The formula of the AM-softmax loss is given by:

$$L_{AM} = -\frac{1}{N} \sum_{i=1}^{N} log \frac{e^{s \cdot \phi(\theta_{y_i})}}{e^{s \cdot \phi(\theta_{y_i})} + \sum_{j=1; j \neq y_i}^{C} e^{s \cdot cos(\theta_j)}} \tag{6}$$

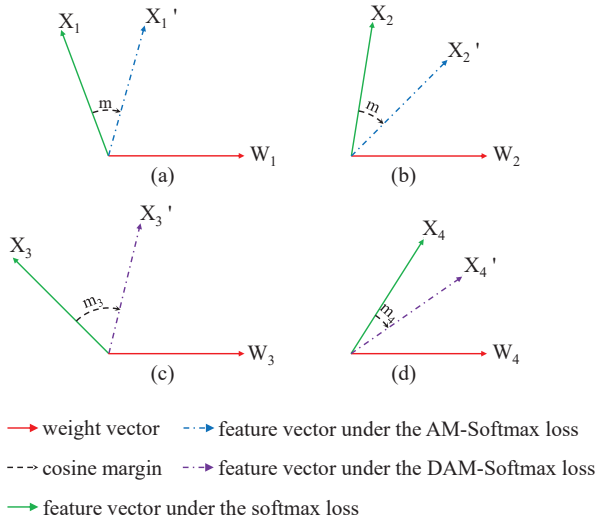


Figure 2: *Illustration of the feature vector and the weight vector under various loss functions. In (a) and (b), $X_1$ and $X_2$ are the feature vectors of two training samples under the conventional softmax loss while $X_1'$ and $X_2'$ are the feature vectors under the AM-Softmax loss, from which we observe that different samples share a constant margin $m$. In (c) and (d), $X_3$ and $X_4$ are the feature vectors of two training samples under the conventional softmax loss while $X_3'$ and $X_4'$ are the feature vectors under the DAM-Softmax loss, where the margin of each sample depends on $cos(\theta)$.*

The cosine margin $m$ is a manually tuned and is usually larger than 0.

## 3. Dynamic-additive-margin softmax loss

As it is used in AM-Softmax loss, the cosine margin is a constant shared by all training samples. It is worth noting that the cosine angle $cos(\theta)$ of different training samples is hardly the same and it changes in the training process, as shown in Figure 1. We propose a dynamic-additive-margin softmax (DAM-Softmax) loss based on the above observation. Our method is based on the assumption that the smaller the $cos(\theta)$ is, the farther the sample is from the corresponding class in the feature space, therefore a larger margin should be set to enforce intra-

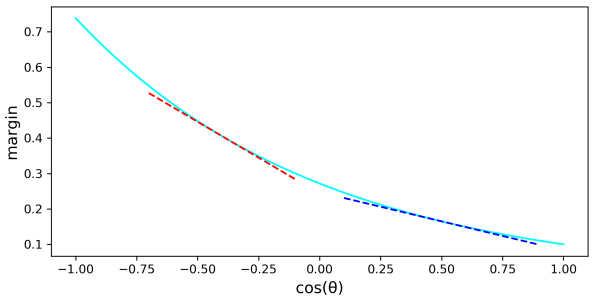


Figure 3: *Corresponding relationship between the margin and the $cos(\theta)$.*

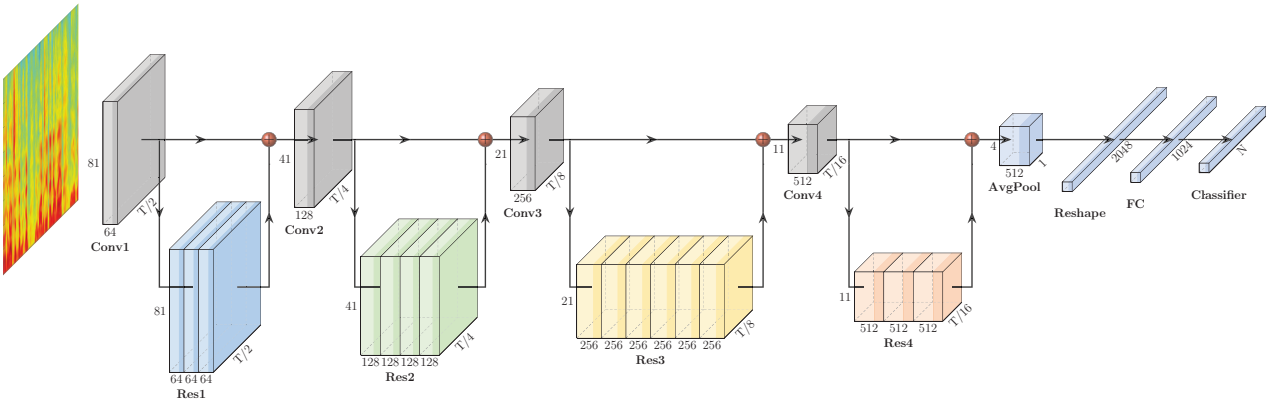


Figure 4: *The architecture of the ResCNN network. The symbol '⊕' represents the element-wise sum. Batch-norm and ReLU activation function are used after each convolutional layer which are omitted for the sake of simplicity. The "Conv'n'" and "Res'n'" labels denote the convolutional layer and the residual module, respectively. The number of slices of the residual module indicates its depth (i.e., the number of residual blocks). For example, 'Res1' consists of 3 residual blocks. The structure of a residual block is shown in Figure 5.*

class compactness. Figure 2 shows the comparison between the AM-Softmax loss and our proposed DAM-Softmax loss. The dynamic margin in the proposed DAM-Softmax loss is defined as:

$$\phi(\theta_{y_i}) = cos(\theta_{y_i}) - m_i \tag{7}$$

$$m_i = \frac{me^{(1-cos(\theta_{y_i}))}}{\lambda} \tag{8}$$

where $m_i$ is the cosine margin of the $i$-th sample, $m$ is the basic margin value, and $\lambda$ is the control factor that controls margin range. Hence, the DAM-Softmax loss function is formulated as:

$$L_{DAM} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\cdot(cos(\theta_{y_i})-m_i)}}{Z} \tag{9}$$

$$Z = e^{s\cdot(cos(\theta_{y_i})-m_i)} + \sum_{j=1;j\neq y_i}^{C} e^{s\cdot cos(\theta_j)} \tag{10}$$

Figure 1 shows that the cosine angle $cos(\theta)$ is relatively small in the initial stage of training and it increases as more epochs are performed. The margin decreases as the $cos(\theta)$ increases, in order to accelerate the margin reduction speed and thus make the training model converge faster, we choose the exponential function as $cos(\theta)$ to margin conversion method in Eq. (8). Figure 3 illustrates the corresponding relationship between the margin and the $cos(\theta)$ when $m$ and $\lambda$ are set to 0.2 and 2 respectively, and the dotted lines indicate the slope of the curve (i.e., the margin reduction speed).

# 4. Experiments

## 4.1. Experimental setup

### *4.1.1. Datasets*

We conduct network training on the development set of VoxCeleb1 (1211 speakers) [3] and VoxCeleb2 (5994 speakers) [8] without any data augmentation, respectively. VoxCeleb1 contains over 100,000 utterances from 1,251 speakers while VoxCeleb2 contains over 1 million utterances from 6,112 speakers. ASV systems are evaluated on the VoxCeleb1 test set,

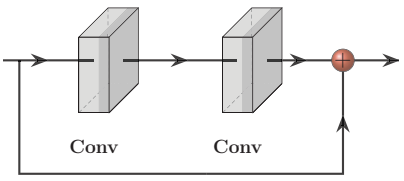


Figure 5: *The structure of a residual block. Here, 'Conv' represents the convolutional layer.*

the extended and hard test sets (VoxCeleb1-E and VoxCeleb1-H, respectively). Notably, the VoxCeleb1 test set consists of 37,720 pairs from 40 speakers, VoxCeleb1-E contains 581,480 pairs from the whole VoxCeleb1 dataset (1251 speakers) and VoxCeleb1-H contains 552,536 pairs that are sampled from speakers with the same gender and nationality. In addition to being evaluated on the VoxCeleb dataset, the systems are further evaluated on the Core condition of the SITW dataset [19] to investigate the performance of our proposed DAM-Softmax loss more comprehensively.

### *4.1.2. Networks*

The residual CNN introduces residual block into the CNN network and has achieved great success in extracting speaker features [21, 22, 23]. Our ResCNN architecture, as shown in Figure 4, accepts $1\times$ 161 $\times T$ spectrogram as its input, in which $T$ denotes the number of frames in the spectrogram. We adopt 4 residual modules with the depths of 3, 4, 6 and 3 respectively, while a $5 \times 5$ filter size, $2 \times 2$ stride convolutional layer is applied to link the residual modules with different channels (i.e., *Conv2*, *Conv3* and *Conv4* in Figure 4). Figure 5 shows the structure of a residual block, which contains two convolutional layers with $3 \times 3$ filters and $1 \times 1$ stride. During training, we randomly sample 3-second utterances from each audio file to generate spectrogram through a hamming window of width 20 ms and step 10 ms, while the full length utterances are used during testing. The *AvgPool* layer is implemented with 2d adaptive average pooling layer, which ensures that the size of the output

Table 1: *Speaker verification performance on the VoxCeleb1 test set, the extended and hard test sets (VoxCeleb1-E and VoxCeleb1-H, repsectively), and the evaluation set of SITW Core.*

| | Front-end model | Loss | Dims | Training set | EER(%) |
|---|---|---|---|---|---|
| **VoxCeleb1 test set** | | | | | |
| Nagrani et al. [3] | i-vector+PLDA | - | - | VoxCeleb1 | 8.80 |
| Nagrani et al. [3] | VGG-M | Softmax | 1024 | VoxCeleb1 | 10.20 |
| Cai et al. [21] | ResNet-34 | A-Softmax + PLDA | 128 | VoxCeleb1 | 4.46 |
| Our implementation | ResCNN | AM-Softmax | 1024 | VoxCeleb1 | 4.65 |
| Proposed | ResCNN | DAM-Softmax | 1024 | VoxCeleb1 | **4.38** |
| Chung et al. [8] | ResNet-34 | Softmax + Contrastive | 512 | VoxCeleb2 | 5.04 |
| Chung et al. [8] | ResNet-50 | Softmax + Contrastive | 512 | VoxCeleb2 | 4.19 |
| Xie et al. [7] | Thin ResNet-34 | Softmax | 512 | VoxCeleb2 | 3.22 |
| Our implementation | ResCNN | AM-Softmax | 1024 | VoxCeleb2 | 2.08 |
| Proposed | ResCNN | DAM-Softmax | 1024 | VoxCeleb2 | **1.94** |
| **VoxCeleb1-E** | | | | | |
| Chung et al. [8] | ResNet-50 | Softmax + Contrastive | 512 | VoxCeleb2 | 4.42 |
| Xie et al. [7] | Thin ResNet-34 | Softmax | 512 | VoxCeleb2 | 3.24 |
| Our implementation | ResCNN | AM-Softmax | 1024 | VoxCeleb2 | 2.28 |
| Proposed | ResCNN | DAM-Softmax | 1024 | VoxCeleb2 | **2.14** |
| **VoxCeleb1-H** | | | | | |
| Chung et al. [8] | ResNet-50 | Softmax + Contrastive | 512 | VoxCeleb2 | 7.33 |
| Xie et al. [7] | Thin ResNet-34 | Softmax | 512 | VoxCeleb2 | 5.17 |
| Our implementation | ResCNN | AM-Softmax | 1024 | VoxCeleb2 | 3.89 |
| Proposed | ResCNN | DAM-Softmax | 1024 | VoxCeleb2 | **3.70** |
| **SITW Core** | | | | | |
| Our implementation | ResCNN | AM-Softmax | 1024 | VoxCeleb2 | 3.96 |
| Proposed | ResCNN | DAM-Softmax | 1024 | VoxCeleb2 | **3.64** |

embedding is the same when test utterances of different lengths are input into the network. The fully connected layer *FC* is employed to yield the 1024-dimensional speaker embedding, and $N$ represents the number of speakers in the training set.

#### *4.1.3. Other Details*

The SGD optimizer with an initial learning rate of 0.1 is employed to optimize the training model. Mini-batch size is 64. We set $m = 0.2$ and $s = 30$ both for the AM-Softmax loss and DAM-Softmax loss, while the control factor is set to 2. The cosine similarity is used as the back-end scoring method.

### 4.2. Experimental results

Experimental results are shown in Table 1, where VoxCeleb1 and VoxCeleb2 refer to their development set, respectively. The proposed DAM-Softmax loss outperforms the AM-Softmax loss in both cases when the training set is VoxCeleb1 or VoxCeleb2 due to the effectiveness of the dynamic margin. More specially, DAM-Softmax loss achieves a 5.8% relative reduction in EER compared with AM-Softmax loss when trained on the VoxCeleb1 and evaluated on the VoxCeleb1 test set, and when the training set is VoxCeleb2, DAM-Softmax loss achieves 6.7%, 6.1%, 4.9% and 8.1% relative reduction in EER compared with AM-Softmax loss when evaluated on the VoxCeleb1 test set, VoxCeleb1-E, VoxCeleb1-H and SITW Core respectively. The results reinforce the idea that dynamically setting the margin for each training sample commensurate with the cosine angle of that sample is more reasonable than using a constant margin shared by all training samples.

In addition, we observe that the performance of our proposed DAM-Softmax loss shows a significant improvement compared with the current state-of-the-art trained on the VoxCeleb2 and evaluated on the VoxCeleb1 test set, VoxCeleb1-E and VoxCeleb1-H (1.94%, 2.14%, and 3.70% in EER, respectively). To the best of our knowledge, our results compare favorably to those reported earlier using the same training and test set.

## 5. Conclusions

In this paper, we proposed a DAM-Softmax loss as an extension to the AM-Softmax loss. In the proposed DAM-Softmax loss, the margin of each training sample dynamically changes during training. This is different from the AM-Softmax loss which uses a constant margin for all training samples. We validated the performance of our method with the ResCNN architecture on the VoxCeleb and SITW datasets. Experimental results show that our proposed DAM-Softmax loss achieves better performance than AM-Softmax loss. Moreover, our results compare favorably to the current state-of-the-art results on the same training and test data.

## 6. Acknowledgements

This work was supported in part by JSPS KAKENHI Grant (20K11883), the National Natural Science Foundation of China under Grant 61771333, the Tianjin Municipal Science and Technology Project under Grant 18ZXZNGX00330.

# 7. References


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

[2] E. Variani, L. Xin, E. Mcdermott, I. L. Moreno, and J. Gonzalez-Dominguez, "Deep neural networks for small footprint text-dependent speaker verification," in *IEEE International Conference on Acoustics*, 2014.

[3] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: a large-scale speaker identification dataset," *Telephony*, vol. 3, pp. 33–039, 2017.

[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] 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.

[6] Y. Tang, G. Ding, J. Huang, X. He, and B. Zhou, "Deep speaker embedding learning with multi-level pooling for text-independent speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6116–6120.

[7] W. Xie, A. Nagrani, J. S. Chung, and A. Zisserman, "Utterance-level aggregation for speaker recognition in the wild," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 5791–5795.

[8] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," *arXiv preprint arXiv:1806.05622*, 2018.

[9] W. Ding and L. He, "Mtgan: Speaker verification through multitasking triplet generative adversarial networks," *arXiv preprint arXiv:1803.09059*, 2018.

[10] G. Bhattacharya, J. Monteiro, J. Alam, and P. Kenny, "Generative adversarial speaker embedding networks for domain robust end-to-end speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6226–6230.

[11] 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*, 2017.

[12] C. Zhang and K. Koishida, "End-to-end text-independent speaker verification with triplet loss on short utterances," *Proc. Interspeech 2017*, pp. 1487–1491, 2017.

[13] W. Liu, Y. Wen, Z. Yu, M. Li, B. Raj, and L. Song, "Sphereface: Deep hypersphere embedding for face recognition," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6738–6746, 2017.

[14] Z. Huang, S. Wang, and K. Yu, "Angular softmax for short-duration text-independent speaker verification," *Proc. Interspeech 2018*, pp. 3623–3627, 2018.

[15] W. Feng, C. Jian, W. Liu, and H. Liu, "Additive margin softmax for face verification," *IEEE Signal Processing Letters*, vol. 25, no. 7, pp. 926–930, 2018.

[16] Y.-Q. Yu, L. Fan, and W.-J. Li, "Ensemble additive margin softmax for speaker verification," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6046–6050.

[17] M. Hajibabaei and D. Dai, "Unified hypersphere embedding for speaker recognition," *arXiv preprint arXiv:1807.08312*, 2018.

[18] Y. Liu, L. He, and J. Liu, "Large margin softmax loss for speaker verification," *arXiv preprint arXiv:1904.03479*, 2019.

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

[20] Y. Li, F. Gao, Z. Ou, and J. Sun, "Angular softmax loss for end-to-end speaker verification," *arXiv preprint arXiv:1806.03464*, 2018.

[21] W. Cai, J. Chen, and M. Li, "Exploring the encoding layer and loss function in end-to-end speaker and language recognition system," *arXiv preprint arXiv:1804.05160*, 2018.

[22] R. Ji, X. Cai, and X. Bo, "An end-to-end text-independent speaker identification system on short utterances," *Proc. Interspeech 2018*, pp. 3628–3632, 2018.

[23] F. Zhao, H. Li, and X. Zhang, "A robust text-independent speaker verification method based on speech separation and deep speaker," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 6101–6105.