INTERSPEECH 2023
20-24 August 2023, Dublin, Ireland

# Dynamic Fully-Connected Layer for Large-Scale Speaker Verification

*Zhida Song*[1,2], *Liang He*[1,2,3]*, *Baowei Zhao*[1,2], *Minqiang Xu, Yu Zheng*

[1]School of Information Science and Engineering, Xinjiang University, China
[2]Xinjiang Key Laboratory of Signal Detection and Processing, China
[3]Department of Electronic Engineering, Tsinghua University, China

szd@stu.xju.edu.cn, heliang@tsinghua.edu.cn

## Abstract

Recently, the mainstream x-vector for speaker verification usually adopts a one-hot encoded fully-connected (FC) layer for classification at the training stage. Suppose a large-scale dataset (e.g., one million speakers) is prepared to optimize the network. The unbearable computation cost and memory requirement are mainly from the FC layer. We propose a dynamic fully-connected (Dynamic FC) layer for speaker verification to achieve a tradeoff between hardware resources and system performance. The proposed Dynamic FC uses a dynamic class queue (DCQ) to store a subset of speaker identity centers and uses an identity-based data loading mechanism to realize memory and time savings. The virtue of the proposed method is that the required memory only depends on the size of the DCQ and does not increase with the number of speakers in the training dataset. The proposed method on the VoxCeleb dataset achieves an EER of 2.345% and a minDCF of 0.261 at a low memory and computation cost.

**Index Terms**: speaker verification, large-scale, dynamic fully-connected layer, dynamic class queue

## 1. Introduction

The speaker verification task is to verify whether a speaker has the target identity by analyzing their acoustic features. In recent years, deep neural networks have made significant progress in the speaker verification task [1, 2, 3]. To further advance the performance and generalizability of speaker verification models, the benefits of self-supervised learning (SSL) in speaker verification tasks are explored in [4, 5, 6, 7]; unsupervised domain adaptation [8, 9, 10] methods aim to transfer the model trained from well-labeled source dataset to the target dataset with weak labels; [11, 12] examined the effect of speaker's different attributes on the speaker verification task; The article [13, 14] makes the distance between intra-class samples smaller by optimizing the loss function; and [15, 16] were devoted to a new framework to solve the problem of adding newly registered users to the model.

In previous studies on the performance and computational complexity of speaker verification tasks, more scholars have focused on the inference phase of the system [17, 18] to obtain lightweight models that can be deployed to resource-constrained devices. For any task related to speaker verification in the training phase, using large-scale speaker verification datasets will improve the system performance and make the model generalize better. However, the large number of speakers causes the number of FC parameters for classification in the model to grow linearly, slows down the model's training, and challenges the hardware resources.

Due to the relatively small number of speakers included in the current open-source datasets in speaker verification, only some scholars have focused on the challenges posed by large-scale datasets during training. Face recognition as a characterization of biometric traits also encounters challenges on large-scale face datasets. In [19, 20], the proposed load-balanced sparse distributed classification algorithm effectively improves the training efficiency. However, this approach does not reduce the number of parameters of the FC layer for classification. So [21] used mapping to make different speakers share weights and used a re-grouping strategy to resolve anchor conflicts. Momentum contrast (MoCo) [22] has achieved great success in the unsupervised domain, where each sample represents a class of ideas. It artificially creates a large-scale problem that can be efficiently solved using a queue and momentum updates. Like MoCo, dynamic class queue [23] works in large-scale face recognition tasks by dynamically storing and updating identity features as a replacement for the FC layer. Further, [24] uses dual loaders to update the dynamic class queue efficiently.

This paper introduces a Dynamic FC layer built upon a contrast learning framework to tackle large-scale speaker verification tasks. We evaluate the effectiveness of our model on the widely-used open-source speaker dataset, VoxCeleb. Furthermore, we compare the Dynamic FC layer and the conventional FC layer, as well as our previously proposed virtual fully-connected (Virtual FC) layer [25], in terms of performance and memory usage. Our main contributions are as follows.

- We propose a Dynamic FC layer, which uses a DCQ to store and update a subset of all speaker identity centers. This approach reduces computation costs and memory requirements in large-scale speaker verification training.
- Instead of using a self-supervision technique, which involves intercepting two equal-length segments of an utterance separately for data enhancement, we utilize an identity-based approach for data loading. This method reduces the training time by half, making it more efficient.

## 2. Virtual Fully-Connected Layer

This section introduces the Virtual FC layer we previously proposed to address large-scale speaker verification tasks.

In the traditional x-vector [26] model, we obtain speaker embeddings of fixed dimensions by an encoder. Then these embeddings are fed to the FC layer with one-hot encoding to get logits. In Virtual FC, we map speakers to $M$ groups ($M$ is a hyperparameter) by taking remainders. During training, the re-grouping strategy is used to resolve the conflicts generated by including the same group of speakers in the mini-batch. As

This work was supported by the National Key R&D Program of China under Grant No. 2022ZD0115801.
* Corresponding author.


10.21437/Interspeech.2023-849

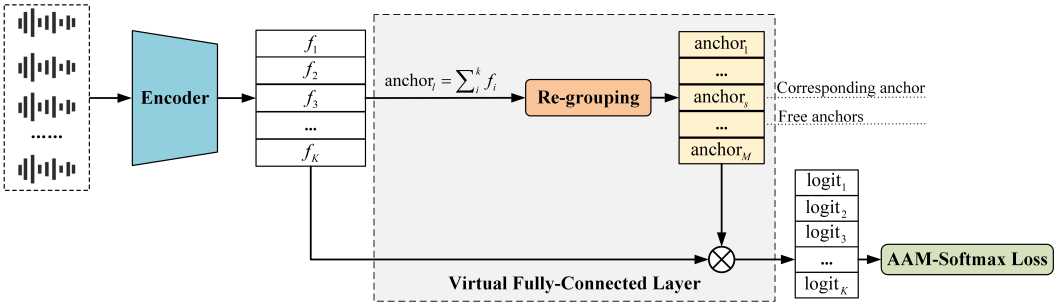


Figure 1: *The pipeline of Virtual FC. $K$ is the number of utterances per speaker in each mini-batch.*

shown in Figure 1, we can set $M$ to be much smaller than the number of speakers to solve the hardware resource constraint problem.

### 2.1. Corresponding Anchor and Free Anchor

When using the Virtual FC layer, we must have more than one speaker utterance in each mini-batch. After the utterances are fed to the encoder and the speaker embedding is obtained, we map speakers to $M$ groups. The identity mapped to the $l$ group shares the $l$-th column parameters of the weight $W$. We call the $l$-th column parameters of $W$ named $anchor_l$. There are two types of weight coefficients in the $W$. One is the corresponding anchor, denoted by $anchor_{corr}$, and the other is the free anchor, denoted by $anchor_{free}$. If there exists an identity in the Virtual FC that belongs to $l$, then $anchor_l$ belongs to $anchor_{corr}$. Otherwise, it belongs to $anchor_{free}$. The type of anchor is dynamically changed during training. The identity first mapped to the $l$ group in the mini-batch will become $anchor_{corr}$. The value of $anchor_{corr}$ is the result of summing the weights of all embeddings of this identity in the mini-batch. The next identities mapped to the $l$ group again will conflict, and we use the re-grouping strategy to temporarily store these features using $anchor_{free}$.

### 2.2. Re-grouping Strategy

Since we are mapping speakers to $M$ groups, there will be cases where different speaker identities will be mapped to one group. Therefore, the corresponding re-grouping strategy is set. There are three cases of identities belonging to group $l$.

1) When $anchor_l$ is not matched, it will change to a $anchor_{corr}$.
2) When $anchor_l$ has been matched, it will be temporarily assigned a $anchor_{free}$, making it a $anchor_{corr}$.
3) The anchor is discarded if there is a conflict and no $anchor_{free}$ exists.

## 3. Dynamic Fully-Connected Layer

In this section, we first introduce an overview of our proposed Dynamic FC layer. Next, we illustrate the principle of the dynamic class queue. Then, we describe the momentum updates that keep the feature in the queue consistent. Finally, it explains how we use the AAM-Softmax loss for backpropagation.

### 3.1. Overview of Dynamic FC

In Virtual FC, we group the speakers by taking the remainder of the mapping size $M$. Speakers in the same group will share the same weight values. This method of randomly clustering different speakers puts a constraint on the performance improvement of the system. To address this problem, we propose a new method called Dynamic FC. As shown in Figure 2, Dynamic FC uses a DCQ to dynamically store a subset of all speaker identity centers based on contrast learning. This method effectively avoids the impact of random clustering of speakers on the system performance.

In Dynamic FC, we introduce twin backbones named Probe Net (P-Net) and Gallery Net (G-Net) [24] to extract speaker features and generate pseudo identity centers. G-Net has the same network structure as P-Net and inherits the parameters from P-Net in a moving average manner. In a mini-batch containing $N$ speakers, we randomly select two utterances for each speaker to get $(x_{i,j}, y_i)$, where $i \in \{1, 2, \cdots, N\}$ and $j \in \{1, 2\}$. $x_{i,1}$ is fed to P-Net to get speaker embeddings $e_{i,1}$, and $x_{i,2}$ is fed to G-Net to generate pseudo identity centers $e_{i,2}$.

As mentioned above, we generate pseudo identity centers through the G-Net network. There are two implications here. One is that, in each training iteration, we want the speaker features extracted by P-Net to be closer to the speaker features extracted by G-Net, so the speaker features extracted by G-Net play the role of speaker identity center. Second, the speaker features extracted by G-Net are different from true identity centers. We gradually approach the true identity centers in the training iterations, so we define the features extracted by G-Net as pseudo identity centers.

The number of utterances in the current open-source dataset is much larger than the number of speakers. Therefore, in contrast to the traditional approach of augmenting features, we use an identity-based approach to generate sample pairs. In self-supervised learning [6, 27], the data is usually passed into the network as two segments of selected speech of fixed length intercepted separately. The data is enhanced in different ways. We also conducted experiments with the same data loading method, and the performance did not improve compared to our proposed method. Our proposed approach of selecting two utterances of a speaker for data loading can save training time. The first benefit is that we do not perform augmentation will reduce the time spent on data pre-processing. Secondly, half of the utterances in each epoch are used for training, reducing the training time by almost half.

### 3.2. Dynamic Class Queue

A queue is defined as a linear data structure that is open at both ends, and the operations are performed in first-in-first-out (FIFO) order. In Dynamic FC, we use a DCQ storage pseudo identity centers that contain speaker labels. At each iteration,

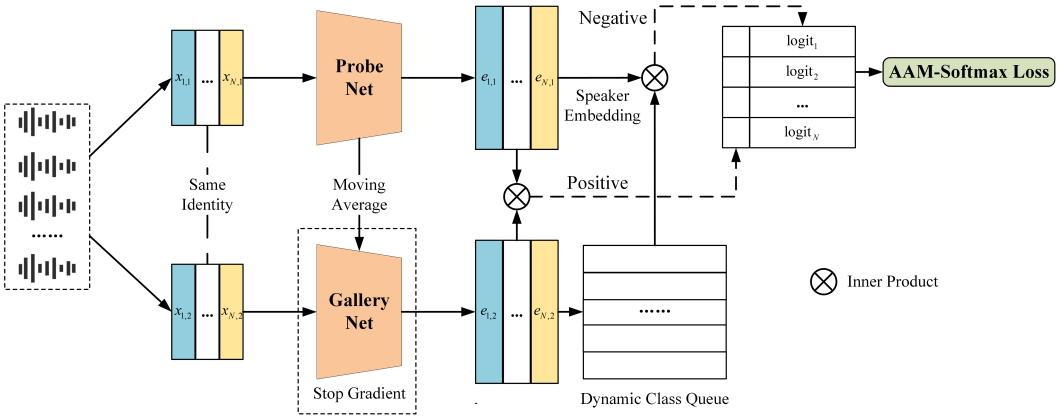


Figure 2: *The pipeline of Dynamic FC. Instead of augmenting the utterance, we randomly select two utterances per speaker in the mini-batch. Then feed one into the P-Net and the other into the G-Net.*

the embeddings in the latest mini-batch enter the queue, and the embeddings in the oldest mini-batch leave the queue.

The queue size $C$ is an integer multiple of the mini-batch and will be set as a hyperparameter. The DCQ stores a subset of all identity centers and will be computed as negative samples for loss. Since no updated parameters are required, the queue size impacts the overall computation time.

### 3.3. Momentum Update

Since a DCQ stores a certain number of pseudo identity centers, it cannot make the queue data perform gradient backpropagation. There is no way for G-Net to update its parameters by backpropagation. A quick update of G-Net parameters will lead to the data in the queue cannot be kept consistent. So we use the moving average to update the parameters in G-Net. We let the parameters of P-Net be $\theta_p$ and the corresponding parameters of G-Net be $\theta_g$. The corresponding equation is as follows.

$$\theta_g \leftarrow \alpha\theta_g + (1-\alpha)\theta_p \tag{1}$$

Where $\alpha \in [0, 1)$ is the momentum coefficient, all values of $\alpha$ are set to 0.999 in our experiments.

### 3.4. AAM-Softmax Loss

Speaker verification tasks usually use the Softmax loss function to classify speakers during training. This paper uses AAM-Softmax [28] as a loss function. The following equation is used.

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s(\cos(\theta_{y_i}+m))}}{e^{s(\cos(\theta_{y_i}+m))}+\sum_{j=1,y_j\neq y_i}^{C} e^{s\cos\theta_{y_j}}} \tag{2}$$

Where $N$ is the number of utterances in each batch fed to P-Net, $C$ is the size of the dynamic class queue, $s$ is a scaling factor and $m$ is a margin between different classes. Specifically, $\cos\theta_{y_i} = \langle e_{i,1}, e_{i,2}\rangle$.

## 4. Experimental setup

### 4.1. Dataset

Our experiments are conducted on the VoxCeleb1&2 [29, 30] datasets. The development set of VoxCeleb2 is used to train models, which contain 5994 speakers. No data augmentation is used. VoxCeleb1 is used as testing data. To evaluate the performance of our system, we used the three official test lists (VoxCeleb1, VoxCeleb1-H, and VoxCeleb1-E). Equal error rate (EER) and minimum detection cost function (minDCF) are used as metrics for the systematic evaluation.

### 4.2. Implementation Details

We use an x-vector structure to extract the speaker embeddings. We set the frame length to 25 ms, and the frame shift to 10 ms. For training, we randomly select 200 frames and use them as input to the model. 80-dimensional Fbanks are used as input features. ResNet34 topology is used for frame-level feature extraction. Then the frame-level features are fed into the attentive statistics pooling (ASP) [31] layer to get speaker embeddings. Every model uses the AAM-Softmax loss to classify speakers. We set the model to train for 25 epochs. Both use the Adam [32] algorithm to optimize the model and decay the learning rate of each parameter group by 0.4 every five epochs.

In our FC experiments, we employ a batch size of 200. For our Dynamic FC experiments, we adopt a larger batch size of 400, wherein each mini-batch comprises 200 speakers. Each speaker has two utterances fed into the P-Net and G-Net networks. We use 512-dimensional speaker embeddings. For the AAM-Softmax loss, we set the margin to 0.3 and the scaling factor to 30. We employ the Adam optimization algorithm with an initial learning rate of 0.01.

The Virtual FC architecture utilizes a batch size of 192, with each mini-batch comprising 64 speakers. Three randomly selected utterances are fed into the training network for each speaker. We use 256-dimensional speaker embeddings. For the AAM-Softmax loss, we set the margin to 0.5 and the scaling factor to 30. We employ the Adam optimization algorithm with an initial learning rate of 0.0002.

## 5. Results and analysis

In Table 1, we compare FC, Virtual FC, and Dynamic FC performance. The size column shows the number of groups to which speakers are mapped in the Virtual FC section, while in the Dynamic FC section, it indicates the queue size of the DCQ. FC achieves the best EER and minDCF results across all three test lists using one-hot encoding.

The Virtual FC layer maps 5994 speakers to 600, 1200, 1800, and 3000 groups. The results show that the system's

Table 1: *EER(%) and minDCF(0.01) results on VoxCeleb1. The number in the size column represents the number of groups to which speakers are mapped in the Virtual FC section, while in the Dynamic FC section, it represents the queue size of the DCQ.*

| Method | Size | VoxCeleb1 | | VoxCeleb1-H | | VoxCeleb1-E | |
|---|---|---|---|---|---|---|---|
| | | EER(%) | minDCF(0.01) | EER(%) | minDCF(0.01) | EER(%) | minDCF(0.01) |
| FC | - | 1.500 | 0.15811 | 2.480 | 0.23547 | 1.459 | 0.15892 |
| Virtual FC | 600 | **2.867** | 0.29951 | 4.805 | 0.41605 | **2.939** | 0.31312 |
| | 1200 | 3.079 | **0.29786** | **4.705** | **0.39518** | 2.976 | **0.30607** |
| | 1800 | 3.510 | 0.33366 | 5.417 | 0.44847 | 3.415 | 0.34464 |
| | 3000 | 4.074 | 0.40232 | 6.307 | 0.49780 | 4.100 | 0.38667 |
| Dynamic FC | 600 | 2.665 | 0.28712 | 4.737 | 0.44772 | 2.708 | 0.32213 |
| | 1200 | 2.803 | 0.28574 | 4.577 | 0.42459 | 2.656 | 0.31060 |
| | 1800 | 2.649 | 0.27994 | 4.652 | 0.43419 | 2.669 | 0.31589 |
| | 3000 | **2.345** | **0.26115** | **4.191** | **0.39696** | **2.394** | **0.28295** |
| | 4200 | 2.457 | 0.28809 | 4.495 | 0.42024 | 2.531 | 0.30313 |

performance improves as the number of mapped identities decreases. The best performance is achieved with 600 and 1200 mapped identities. In model training, we randomly bring the features of one to two people closer together when dividing into 3000 groups and randomly bring the features of four to five people closer together when dividing into 600 groups. Higher numbers of people result in greater error tolerance. When selecting a group of two people at random, there is a high likelihood that the features of the two speakers within the group are far apart in space.

When the size of the DCQ was set to 3000, the Dynamic FC layer achieved the best performance on all three test sets. Using about 3000 identity centers can classify the VoxCeleb2 dataset of 5994 speakers for classification. From the data in the table, we know that the performance of the DCQ queue size to obtain the median value is better than the other cases. We analyze the reasons for this situation from two aspects. One is that the development set of VoxCeleb2 contains 5994 speakers. When setting a smaller queue, it is difficult to carry the information of all the speakers in the dataset; second is that when the queue size keeps getting larger, a more binding model or a larger dataset (e.g., a dataset of a million speakers) is needed to distinguish the speakers. In our experiments, we set the margin in the AAM-Softmax loss differently from the usual setting of 0.2. We set larger values of 0.3 and 0.5. In addition to the loss function, other aspects of increasing the constraint are further worth exploring. Similar to speaker recognition in face recognition, large-scale task datasets are usually trained with 1% of identity centers selected from millions of identity datasets, which we believe is why the performance does not improve further as the cohort size increases.

Dynamic FC outperforms Virtual FC in most test sets when $M$ and $C$ are set to the same value. Specifically, when the value is set to 3000, Dynamic FC significantly improves the EER and minDCF performance on VoxCeleb1 by 33% and 22%, respectively, compared to Virtual FC. This improvement is mainly attributed to the performance constraint imposed by Virtual FC, which forces different speakers to share weights. On the other hand, Dynamic FC stores a subset of all speaker identity centers, which are dynamically changing and becoming more accurate as the model is trained.

The FC layer is not suitable for large-scale speaker verification tasks. Typically, one-hot encoding is used, which results in the number of parameters growing linearly with the number of speakers. In Table 2, we analyze the number of parameters for different FC layers when the size is 600. When the speaker embeddings are 512-dimensional, the memory usage of the FC layer is approximately ten times that of the Dynamic FC layer. For the Virtual FC layer that uses 256-dimensional speaker embeddings, its memory usage is approximately one-twentieth that of the FC layer. It is conceivable that when a dataset of one million speakers is used, choosing values much smaller than the number of speakers $M$ or $C$ would solve the hardware resource constraint problem. Additionally, we believe the performance gap can be attributed to using smaller speaker datasets.

Table 2: *The number of parameters for different FC layers when the size is set to 600.*

| Method | Wight Shape | # Params | Memory Saving |
|---|---|---|---|
| FC | $512 \times 5994$ | 3069.928K | $1\times$ |
| Virtual FC | $256 \times 600$ | 153.6K | $\sim 20\times$ |
| Dynamic FC | $512 \times 600$ | 307.2K | $\sim 10\times$ |

## 6. conclusion

This paper proposes a Dynamic FC layer for large-scale speaker verification. The Dynamic FC layer utilizes a DCQ to store a subset of all speaker identity centers during training and loads data based on identity for efficient time and cost savings. Additionally, we compare this approach with our previously proposed Virtual FC layer. Our experiments on the VoxCeleb datasets demonstrate that a DCQ (Dynamic FC) yields better performance than a group of speakers sharing weights (Virtual FC). In terms of performance, a gap exists between our proposed approach and the one-hot coding-based FC layer. However, our article aims to provide a solution for large-scale speaker verification tasks rather than simply striving for better results than FC layers. Furthermore, the Dynamic FC layer is suitable for larger speaker datasets, and we will further validate its efficacy.

# 7. References


[1] B. Desplanques, J. Thienpondt, and K. Demuynck, "ECAPA-TDNN: Emphasized Channel Attention, Propagation and Aggregation in TDNN Based Speaker Verification," in *Proc. Interspeech 2020*, 2020, pp. 3830–3834.

[2] Y.-J. Zhang, Y.-W. Wang, C.-P. Chen, C.-L. Lu, and B.-C. Chan, "Improving Time Delay Neural Network Based Speaker Recognition with Convolutional Block and Feature Aggregation Methods," in *Proc. Interspeech 2021*, 2021, pp. 76–80.

[3] Y. Zhang, Z. Lv, H. Wu, S. Zhang, P. Hu, Z. Wu, H. yi Lee, and H. Meng, "MFA-Conformer: Multi-scale Feature Aggregation Conformer for Automatic Speaker Verification," in *Proc. Interspeech 2022*, 2022, pp. 306–310.

[4] N. Vaessen and D. A. Van Leeuwen, "Fine-tuning wav2vec2 for speaker recognition," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7967–7971.

[5] Z. Chen, S. Chen, Y. Wu, Y. Qian, C. Wang, S. Liu, Y. Qian, and M. Zeng, "Large-scale self-supervised speech representation learning for automatic speaker verification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6147–6151.

[6] R. Tao, K. A. Lee, R. K. Das, V. Hautamäki, and H. Li, "Self-supervised speaker recognition with loss-gated learning," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 6142–6146.

[7] S. Chen, Y. Wu, C. Wang, S. Liu, Z. Chen, P. Wang, G. Liu, J. Li, J. Wu, X. Yu, and F. Wei, "Why does Self-Supervised Learning for Speech Recognition Benefit Speaker Recognition?" in *Proc. Interspeech 2022*, 2022, pp. 3699–3703.

[8] H.-R. Hu, Y. Song, L.-R. Dai, I. McLoughlin, and L. Liu, "Class-Aware Distribution Alignment based Unsupervised Domain Adaptation for Speaker Verification," in *Proc. Interspeech 2022*, 2022, pp. 3689–3693.

[9] A. Mccree, S. Shum, D. Reynolds, and D. Garcia-Romero, "Unsupervised Clustering Approaches for Domain Adaptation in Speaker Recognition Systems," in *Proc. The Speaker and Language Recognition Workshop (Odyssey 2014)*, 2014, pp. 265–272.

[10] G. Bhattacharya, J. Alam, and P. Kenny, "Adapting end-to-end neural speaker verification to new languages and recording conditions with adversarial training," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2019, pp. 6041–6045.

[11] C. Luu, S. Renals, and P. Bell, "Investigating the contribution of speaker attributes to speaker separability using disentangled speaker representations," in *Proc. Interspeech 2022*, 2022, pp. 610–614.

[12] X. Qin, N. Li, W. Chao, D. Su, and M. Li, "Cross-Age Speaker Verification: Learning Age-Invariant Speaker Embeddings," in *Proc. Interspeech 2022*, 2022, pp. 1436–1440.

[13] L. Li, R. Nai, and D. Wang, "Real additive margin softmax for speaker verification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7527–7531.

[14] L. Ruida, F. Shuo, M. Chenguang, and L. Liang, "Adaptive Rectangle Loss for Speaker Verification," in *Proc. Interspeech 2022*, 2022, pp. 301–305.

[15] E. Yoo, H. Song, T. Kim, and C. Lee, "Online Learning of Open-set Speaker Identification by Active User-registration," in *Proc. Interspeech 2022*, 2022, pp. 5065–5069.

[16] S. Yang, D. Das, J. Cho, H. Park, and S. Yun, "Domain Agnostic Few-shot Learning for Speaker Verification," in *Proc. Interspeech 2022*, 2022, pp. 595–599.

[17] Q. Li, L. Yang, X. Wang, X. Qin, J. Wang, and M. Li, "Towards lightweight applications: Asymmetric enroll-verify structure for speaker verification," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 7067–7071.

[18] Y. Wei, J. Du, H. Liu, and Q. Wang, "CTFALite: Lightweight Channel-specific Temporal and Frequency Attention Mechanism for Enhancing the Speaker Embedding Extractor," in *Proc. Interspeech 2022*, 2022, pp. 341–345.

[19] X. An, X. Zhu, Y. Gao, Y. Xiao, Y. Zhao, Z. Feng, L. Wu, B. Qin, M. Zhang, D. Zhang *et al.*, "Partial fc: Training 10 million identities on a single machine," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1445–1449.

[20] X. An, J. Deng, J. Guo, Z. Feng, X. Zhu, J. Yang, and T. Liu, "Killing two birds with one stone: Efficient and robust training of face recognition cnns by partial fc," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4042–4051.

[21] P. Li, B. Wang, and L. Zhang, "Virtual fully-connected layer: Training a large-scale face recognition dataset with limited computational resources," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 315–13 324.

[22] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 9729–9738.

[23] B. Li, T. Xi, G. Zhang, H. Feng, J. Han, J. Liu, E. Ding, and W. Liu, "Dynamic class queue for large scale face recognition in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 3763–3772.

[24] K. Wang, S. Wang, P. Zhang, Z. Zhou, Z. Zhu, X. Wang, X. Peng, B. Sun, H. Li, and Y. You, "An efficient training approach for very large scale face recognition," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4083–4092.

[25] Z. Song, L. He, Z. Fang, Y. Hu, and H. Huang, "Virtual fully-connected layer for a large-scale speaker verification dataset," in *Biometric Recognition: 16th Chinese Conference, CCBR 2022, Beijing, China, November 11–13, 2022, Proceedings*. Springer, 2022, pp. 382–390.

[26] 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.

[27] T. Lepage and R. Dehak, "Label-Efficient Self-Supervised Speaker Verification With Information Maximization and Contrastive Learning," in *Proc. Interspeech 2022*, 2022, pp. 4018–4022.

[28] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 4690–4699.

[29] A. Nagrani, J. S. Chung, and A. Zisserman, "Voxceleb: A large-scale speaker identification dataset," in *Proc. Interspeech 2017*, 2017, pp. 2616–2620. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2017-950

[30] J. S. Chung, A. Nagrani, and A. Zisserman, "Voxceleb2: Deep speaker recognition," in *Proc. Interspeech 2018*, 2018, pp. 1086–1090. [Online]. Available: http://dx.doi.org/10.21437/Interspeech.2018-1929

[31] K. Okabe, T. Koshinaka, and K. Shinoda, "Attentive Statistics Pooling for Deep Speaker Embedding," in *Proc. Interspeech 2018*, 2018, pp. 2252–2256.

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