EURASIP Journal on Advances in Signal Processing

RESEARCH Open Access

Text-independent speaker recognition based on adaptive course learning loss and deep residual network

Qinghua Zhong[1,2*], Ruining Dai[1], Han Zhang[1], Yongsheng Zhu[1] and Guofu Zhou[2]

*Correspondence:
zhongqinghua@m.scnu.edu.cn
[1]School of Physics and Telecommunication Engineering, South China Normal University, Guangzhou 510006, China
[2]South China Academy of Advanced Optoelectronics, South China Normal University, Guangzhou 510006, China

Abstract
Text-independent speaker recognition is widely used in identity recognition that has a wide spectrum of applications, such as criminal investigation, payment certification, and interest-based customer services. In order to improve the recognition ability of log filter bank feature vectors, a method of text-independent speaker recognition based on deep residual networks model was proposed in this paper. The deep residual network was composed of a residual network (ResNet) and a convolutional attention statistics pooling (CASP) layer. The CASP layer could aggregate frame-level features from the ResNet into an utterance-level features. Extracting speech features for each speaker using deep residual networks was a promising direction to explore, and a straightforward solution was to train the discriminative feature extraction network by using a margin-based loss function. However, a margin-based loss function often has certain limitations, such as the margins between different categories were set to be the same and fixed. Thus, we used an adaptive curriculum learning loss (ACLL) to address the problem and introduce two different margin-based losses for this problem, i.e., AM-Softmax and AAM-Softmax. The proposed method was applied to a large-scale VoxCeleb2 dataset for extensive text-independent speaker recognition experiments, and average equal error rate (EER) could achieve 1.76% on VoxCeleb1 test dataset, 1.91% on VoxCeleb1-E test dataset, and 3.24% on VoxCeleb1-H test dataset. Compared with related speaker recognition methods, EER was improved by 1.11% on VoxCeleb1 test dataset, 1.04% on VoxCeleb1-E test dataset, and 1.69% on VoxCeleb1-H test dataset.

Keywords: Text-independent, Speaker recognition, Adaptive curriculum learning loss, Deep residual network, Convolutional attention statistics pooling

1 Introduction

Speaker recognition (SR) [1] is the process of automatically recognizing a speaker based on original speech samples. It has become an increasingly important technology of recognizing identities in many electronic intelligent applications, law enforcement, and forensics [2, 3]. Speaker recognition includes speaker verification (SV) and speaker identification (SI) [4], and speaker recognition can be categorized into text-dependent speaker recognition (TD-SR) [5] and text-independent speaker recognition (TI-SR) [6]. The SV aims to verify whether a speech belongs to a specific enrolled speech, while the SI aims to classify the identification of an unknown speech among a specific set of enrolled speech. For the TD-SR system, the speech text during training must be identical to the speech text during testing. By contrast, for the TI-SR system, the speaker recognition process does not depend on the speech text being spoken by the speaker. Therefore, the TI-SR case is more difficult than the TD-SR case due to larger variations introduced by different speech transcriptions and duration. In the paper, our work focuses on the TI-SR case with respect to speaker recognition tasks, since it is more challenging and has greater practical significance.

Generally, speaker recognition tasks based the TI-SR system usually follow a similar three stage pipeline: (i) frame-level feature vectors extraction, (ii) temporal aggregation of frame-level feature vectors, and (iii) optimization of a classification loss. Frame-level feature vectors extraction processing can be achieved by using the backbone CNN structure, which is usually a 2D CNN with convolution in time domain and frequency domain [5, 7, 8]. Utterance-level processing forms speaker representation based on the frame-level output. A pooling layer is used to aggregate frame-level information to form utterance-level representation. For the TD-SR system, all test utterance of the speaker were preseted in a training dataset, so the TD-SR system was equivalent to one-to-one verification, which could be regarded as a classification problems [5]. For the TI-SR system, the test dataset and training dataset were disjoint. Therefore, the feature vectors of a speaker needed to be projected into a discriminative embedding space, which could be treated as a metric learning problem [7]. Generally, a research method based on TI-SR case was mainly realized by the original softmax loss function. However, for text-independent metric learning problems, the discriminativeness of learning features was not enough such as the triplet loss [8, 9].

Recently, researchers have used several margin-based loss functions to carry out speaker recognition experiments and have obtained competitive results. For example, A-Softmax [10], AM-Softmax [11, 12], and AAM-Softmax [13] could significantly increase the margins of different categories. Therefore, a powerful speaker recognition deep network was proposed, based on a GhostVLAD layer and a AM-Softmax that was used to aggregate "thin-ResNet" architecture frame features [12]. However, the margins between different categories were set to be the same and fixed, which could not be well adapted to various situations. For example, the AM-Softmax and AAM-Softmax loss functions required extensive experiments to tune the two dependent super-parameters to find the optimal values.

In addition, a clustering distance loss algorithm directly reduced intra-class variation and expanded the margins between different categories [14]. Recently, researchers have used temporal averaging pooling (TAP) to aggregate frame-level features, and an utterance-level features representation was formed by averaging all frame-level feature vectors [15]. However, these methods do not distinguish speech samples well. Thus, an attention mechanism was introduced to aggregate the frame-level features in deep learning model [14, 16]. By assigning different weights to different utterance samples, this allows the weights to be focused on the important features. In addition, a higher-order statistics were introduced into the field of speaker recognition to calculate the mean and standard deviation of frame-level features [17]. Furthermore, an attention mechanism and statistical methods were combined to propose an attention statistical pooling (ASP) [18].

It provided an importance weighted standard deviation and weighted average of speaker features and calculated the sample weight importance by an attention mechanism.

Therefore, a method based on an ACLL and a deep residual network was proposed for TI-SR system in this paper, and the method realized a speaker recognition training strategy. The deep residual network named Res-CASP had good time modeling ability, and it could extract effective information of speech feature vectors. The CASP aggregated frame-level features of the deep residual network to form an utterance-level features. The ACLL was a loss function which optimized speaker features. Speaker features were extracted and fed into the ACLL based a deep residual network for text-independent speaker recognition.

2 Overall framework

As shown in Fig. 1, the overall framework consists of three parts: speech feature vectors extraction, deep residual network Res-CASP, and ACLL. Feature vectors extraction is used to convert the original speech into 64-dimensional log filter bank feature vectors. The Res-CASP framework includes a ResNet, a CASP layer, and a fully connected (FC) layer. The CASP layer aggregates frame-level features of the ResNet into utterance-level features, and the FC layer constrains the utterance-level feature vectors to 512-dimensional vector representation. The ACLL is the feature vectors which optimizes the output of the Res-CASP framework. The trained Res-CASP model is used for the final speaker recognition.

2.1 Log filter bank features extraction

The original speech signal is a one-dimensional time domain signal, and the input of deep residual network is a two-dimensional signal data. Generally, there are two main ways to extract features for speech: MFCC [19] and log filter bank [20]. Because MFCC is based on log filter bank, the feature extraction of log filter bank is more in line with the essence of speech signal, fitting the characteristics of human ear reception, and MFCC does DCT

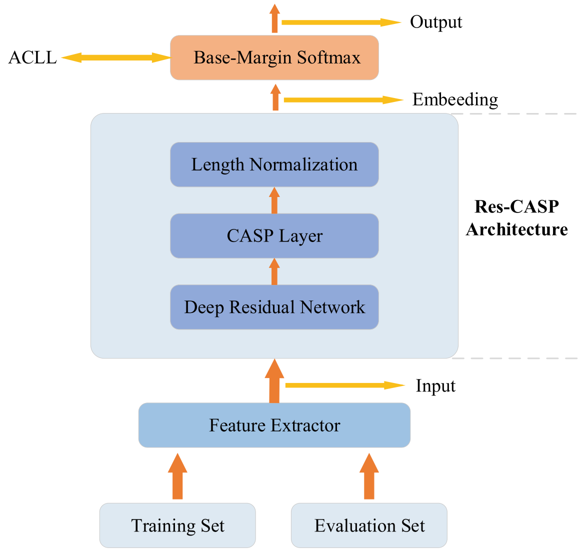


Fig. 1 The proposed text-independent speaker recognition framework

decorrelation processing on log filter bank, so log filter bank contains more information than MFCC. Therefore, the original speech signal is first extracted as a log filter bank feature vectors. The specific steps of log filter bank feature vectors extraction [19, 20] are as follows:

1) Pre-emphasis

Pre-emphasis is a high-pass filter whose purpose is to boost high-frequency signal components. In terms of acoustic features extraction, the pre-emphasis filter is shown in Eq. 1.

$$H(z) = 1 - \alpha z^{-1} \tag{1}$$

where α is a pre-emphasis coefficient, and z is an input signal of original speech.

2) Framing

By dividing the speech signal into shorter frames, the signal can be regarded as a steady-state signal in each frame, which can be processed as the steady-state signal in the same way. At the same time, in order to make parameters between two adjacent frames more smoothly, there is a partial overlap.

3) Windowing function

The purpose of the windowing function is to reduce the leakage in the frequency domain. Each frame of speech signal is multiplied by a Hamming window with a frame length and a frame shift [19]. Each frame signal after preprocessing is multiplied by the Hamming window to increase the continuity of the frame. The calculation process is shown in Eq. 2.

$$T(n) = S(n) \times (0.54 - 0.46cos[\,2\pi n/(N-1)]\,),\ 0 \le n \le N-1 \tag{2}$$

where $S(n)$ is the input of speech signal after pre-emphasis and framing, and N is a frame length.

4) Fast Fourier transform

Then, each frame of speech signal is performed with fast Fourier transform, and the time domain data is converted into frequency domain data. As shown in Eq. 3.

$$X(k) = \sum_{n=0}^{N-1} T(n) e^{-2\pi nk/P},\ 0 \le k \le P \tag{3}$$

where $T(n)$ is an input speech signal after windowing function, P is the number of Fourier transform points, and k is the frequency index ($k = 0, 1, 2, ..., P-1$).

5) Energy calculation of mel filter banks

The energy spectrum is fed to several triangular bandpass filters $H_m(k)$. Each filter has triangular filtering characteristics [18]. In the frequency domain, the energy spectrum $|X(k)|^2$ and the frequency domain response $H_m(k)$ are multiplied and added. The calculation process is shown in Eq. 4.

$$M(l) = \sum_{k=0}^{N-1} |X(k)|^2 H_m(k) \tag{4}$$

where $X(k)$ is the signal after fast fourier transform, and $H_m(k)$ is the triangular band-pass filter. Its frequency response is shown in Eq. 5.

$$H_m(k) = \begin{cases} 0, & k < f(m-1) \\ \frac{k-f(m-1)}{f(m)-f(m-1)}, & f(m-1) \leq k < f(m) \\ \frac{f(m+1)-k}{f(m+1)-f(m)}, & f(m) < k \leq f(m+1) \\ 0, & k > f(m+1) \end{cases} \tag{5}$$

where $f(m)$ is the center frequency of $H_m(k)$, $0 \leq m < L$, and L is the number of bandpass filters.

6) Log energy spectrum

For the mth frame, the log energy spectrum of filter is defined as Eq. 6.

$$e(l) = log(M(l)) \tag{6}$$

where $M(l)$ is the feature vectors after calculating energy of the mel filter.

2.2 Structure of residual block

ResNet is a way to alleviate the difficulty of training deep convolutional neural network [21, 22]. It is learning the following layers of deep network into identity mapping, that is, $h(x) = x$, so the model degenerates into shallow network. The identity mapping greatly reduces the number of training parameters in the neural network. The ResNet is composed of many stacked residual blocks (Res-Blocks). The structure of Res-Blocks is shown in Fig. 2. The Res-Blocks is composed of two 2D convolutional layers. The identity mapping can be used to map each Res-Blocks input feature vectors to an output feature vectors. The expression defined by Res-Block as shown in Eq. 7.

$$\Gamma = F(x, W_i) + x \tag{7}$$

where x and Γ are input feature vectors and output feature vectors, respectively. W_i is the learnable weight, and $F(x, W_i)$ is the output of residual mapping. In addition, the identity mapping connection of does not add additional parameters and computational complexity.

In order to make full use of feature learning capabilities of ResNet and reduce loss of feature information, we use identity mapping to reduce data dimensions. In addition, in each convolutional layer, the stride is set to 1, the padding is set to SAME, and zero padding is used to prevent information from being lost at the edge of the cube.

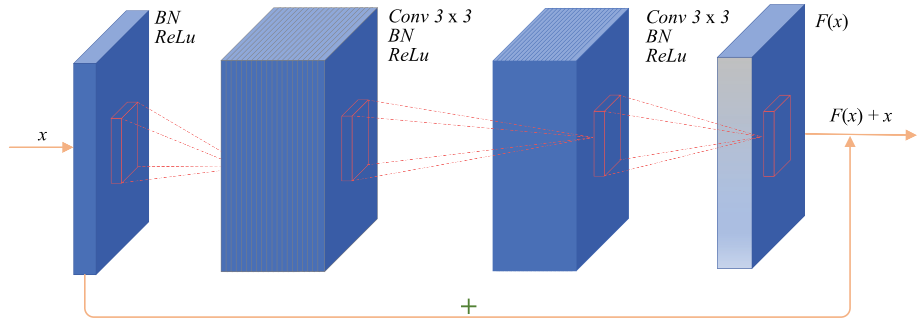


Fig. 2 Flow chart of residual block construction

2.3 Structure of CASP

By combining higher-order statistics and attention mechanism, the ASP is proposed [18]. It provides importance weighted standard deviations as well as the weighted means of frame-level features, for which the importance is calculated by an attention mechanism. Such previous work, however, has been evaluated only in such limited tasks as fixed-duration text-independent [18, 23]. Therefore, we propose a new pooling method, called CASP. The CASP is used to aggregate the frame-level features of the deep residual network model to form utterance-level features. This enables speaker embedding to more accurately and efficiently capture speaker factors with respect to long-term variations. The calculation process of CASP layer is shown in Fig. 3.

Firstly, the frame-level feature vectors $\{x_1, x_2, ..., x_T\}$ of the deep residual network are projected onto one-dimensional convolutional layers to obtain the abstract feature vectors on hidden unit $\{h_1, h_2, ..., h_T\}$.

Secondly, the score is normalized over all frames by a softmax function, which indicates relative importance of the hidden unit. The weight calculation formula for each sample is shown in Eq. 8.

$$w_t = \frac{exp(h_t)}{\sum_{t=1}^{T} exp(h_t)} \tag{8}$$

where h_t is the input feature vectors, and w_t is the weight ratio of each feature vector.

Therefore, utterance-level features can be expressed by weighted sum of frame-level features, and the calculation formula of the weighted sum is shown in Eq. 9.

$$e_t = \sum_{t}^{T} w_t x_t \tag{9}$$

where x_t is the input of feature vectors, and the normalized score e_t is then used as the weight in the pooling layer to calculate the weighted mean vector.

Finally, higher-order statistics with the attention mechanism are combined, that is, CASP. It can generate the mean and standard deviation by attention mechanism. Therefore, the weighted standard deviation is defined as Eq. 10.

$$\sigma = \sqrt{\sum_{t}^{T} w_t x_t \odot x_t - e_t \odot e_t} \tag{10}$$

where σ is the weighted standard deviation, and the advantages of higher-order statistics and attention mechanisms are applied to the weighted standard deviation.

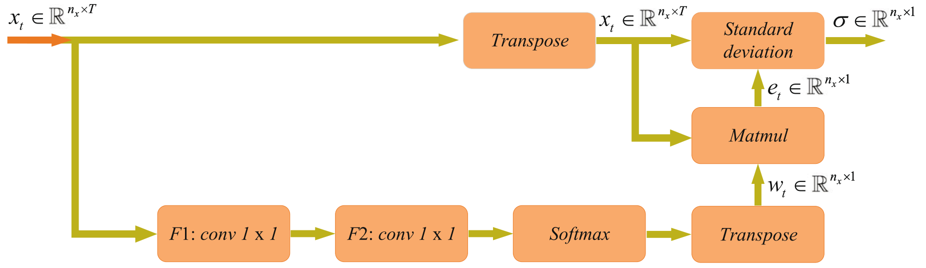

$x_t \in \mathbb{R}^{n_x \times T}$
Transpose
$x_t \in \mathbb{R}^{n_x \times T}$
Standard deviation
$\sigma \in \mathbb{R}^{n_x \times 1}$
$e_t \in \mathbb{R}^{n_x \times 1}$
Matmul
$w_t \in \mathbb{R}^{n_x \times 1}$
F1: conv 1 x 1
F2: conv 1 x 1
Softmax
Transpose

Fig. 3 The calculation diagram of the CASP layer

2.4 Structure of ACLL

Loss function design is pivotal for large-scale speaker recognition. Current state-of-the-art deep speaker recognition methods mostly adopt softmax-based classification loss [12]. Since the learned features with the original softmax loss are not guaranteed to be discriminative enough for practical speaker recognition problem, margin-based losses [24–26] are proposed. Though the margin-based loss functions are verified to obtain good performance, they do not take the difficultness of each sample into consideration, while ACLL emphasizes easy samples first and hard samples later, which is more reasonable and effective. The original softmax loss is formulated as follows:

$$L1 = -\sum_{i=1}^{N} log \frac{e^{W_{y_i} x_i + b_{y_i}}}{\sum_{j=1}^{n} e^{W_j x_i + b_j}} \tag{11}$$

where $x_i \in R^d$ denotes the deep feature of ith sample which belongs to the y_i class, $W_j \in R^d$ denotes the jth column of the weight $W \in R^{d\times n}$, and b_j is the bias term. The class number and the embedding feature size are n and d, respectively. In practice, the bias is usually set to $b_j = 0$ and the individual weight is set to $\| W_j \| = 1$ by l_2 normalization. The deep feature is also normalized and re-scaled to s. Thus, the original softmax can be modified as follows:

$$L = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s\cdot\varphi(cos\theta_{y_i})}}{e^{s\cdot\varphi(cos\theta_{y_i})} + \sum_{j\neq y_i} e^{s\cdot Y(t,cos\theta_j)}} \tag{12}$$

where $\varphi(cos\theta_{y_i})$ and $Y(t, cos\theta_j)$ are adjusted for the similarity of the positive and negative cosine, respectively. $cos\theta$ is the cosine similarity of input feature vector y_i and weight w_i, s is the coefficient which can increase recognition speed of model, and N is the total number of classified samples. In the margin-based loss function, such as AM-Softmax [24], such that $\varphi(cos\theta_{y_i}) = cos\theta_{y_i} + m$ and $Y(t, cos\theta_j) = cos\theta_j$; AAM-Softmax [25], such that $\varphi(cos\theta_{y_i}) = cos(\theta_{y_i} + m)$, $Y(t, cos\theta_j) = cos\theta_j$. However, it only modifies the sine and cosine similarity of each sample to enhance feature discrimination, it could not adapt to various situations.

Therefore, ACLL is proposed [27]. The ACLL is defined as Eq. 13.

$$L_{ACLL} = -\frac{1}{N}\sum_{i=1}^{N} log \frac{e^{s\cdot\varphi(t,cos\theta_{y_i})}}{e^{s\cdot\varphi(t,cos\theta_{y_i})} + \sum_{j\neq y_i} e^{s\cdot Y(t^{(k)},cos\theta_j)}} \tag{13}$$

where $\varphi(t, cos\theta_{y_i})$ and $Y(t, cos\theta_j)$ are defined by Eqs. 14 and 15, respectively, and s is a scaling factor of deep feature vectors. It should be noted that the positive cosine similarity can adopt any margin-based loss functions, and here, we adopt AAM-Softmax as an example. In the early training stage, learning from easy samples is beneficial to model convergence. Thus, t should be close to zero and $I(\cdot) = t + cos\theta_i$ is smaller than 1. Therefore, the weights of hard samples are reduced, and easy samples are emphasized relatively. As training goes on, the model gradually focuses on the hard samples, i.e., the value of t shall increase and $I(\cdot)$ is larger than 1. Thus, the hard samples are emphasized with larger weights. Moreover, within the same training stage, $I(\cdot)$ is monotonically decreasing with θ_j so that harder sample can be assigned with larger coefficient according to its difficultness. The value of the parameter t is automatically estimated in the ACLL; otherwise, it may require lots of efforts for manual tuning. Therefore, it can adaptively adjust the relative importance of simple and difficult samples.

$$\varphi(cos\theta_{y_i}) = cos\theta_{y_i} + m \tag{14}$$

where m is the feature margin between different categories, θ_{y_i} is the angle between the feature vectors y_i and the weight w_i , and t is the adaptive estimation parameter.

$$Y(t, cos\theta_j) = \begin{cases} cos\theta_j, & \varphi(cos\theta_{y_i}) \geq cos\theta_j \\ cos\theta_j(t + cos\theta_j), & \varphi(cos\theta_{y_i}) < cos\theta_j \end{cases} \tag{15}$$

where t is adaptive estimation parameters, and exponential moving average (EMA) is used to achieve adaptive parameters. The process is shown as Eq. 16.

$$t^{(k)} = \alpha r^{(k)} + (1 - \alpha)t^{(k-1)} \tag{16}$$

where $r^{(k)}$ is the mean values of the cosine similarity of the kth batch. With the EMA, we avoid the hyper-parameter tuning and make the modulation coefficients of hard sample negative cosine similarities $I(\cdot)$ adaptive to the current training stage.

As shown in Fig. 4, decision conditions are from $cos\theta_{y_i} = cos\theta_j$ (blue line) to $cos(\theta_{y_i} + m) = cos\theta_j$ (yellow line). ACLL is applied to adaptively adjust the weights of difficult samples and the decision condition becomes $cos(\theta_{y_i} + m) = (t + cos\theta_j)cos\theta_j$ (green line). During the training process, the decision boundary of difficult samples changes from a green line (early stage) to another green line (later stage). Simple samples are emphasized first, and then difficult samples are emphasized. In addition, the AAM-Softmax is used as the similarity of sine and cosine, namely $\varphi(t, cos\theta_{y_i}) = cos(\theta_{y_i} + m)$. It can be seen from Eq. 14 that let $Y(t, cos\theta_j) = cos\theta_j$ at the beginning of training.

2.5 Evaluation indicators

We evaluate the framework with equal error rate (EER). EER is denoted by the false rejection (FR) rate equal to the false acceptance (FA) rate, where FR is a correct signal which is recognized as a wrong signal; FA is a wrong signal which is recognized as a correct signal. Definitions of FR rate and FA rate are shown in Eqs. 16 and 17.

$$I_{FR} = \frac{N_{FR}}{N_{Target}} \tag{17}$$

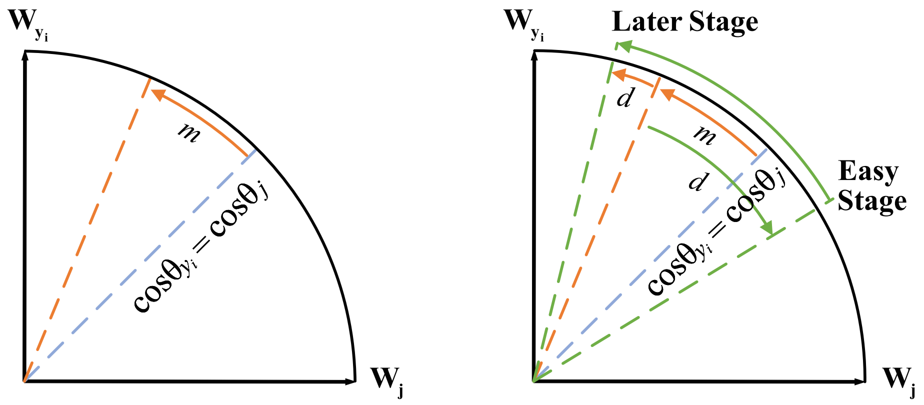

W_{y_i}
m
$cos\theta_{y_i} = cos\theta_j$
W_j
W_{y_i}
Later Stage
d
m
d
$cos\theta_{y_i} = cos\theta_j$
Easy Stage
W_j

Fig. 4 The blue line, yellow line, and green line are Softmax, AAM-Softmax,, and our decision boundary, respectively. *m* is the feature margin of AAM-Softmax. *d* is adaptive feature margin ranges

Table 1 Specific experimental environment

Name	Version
CPU	Intel Xeon(R) Gold 5218 CPU @2.30 GHz ×64
GPU	NVIDIA GeForce 2*RTX 2080Ti 11 GB
RAM	RDIMM 64 GB
OS	Ubuntu 18.04.2 LTS
Frameworks	Pytorch-GPU 1.14.0

where N_{FR} is the number of false rejections, and N_{Target} is the total number of real evaluations.

$$I_{FA} = \frac{N_{FA}}{N_{impostor}} \tag{18}$$

where N_{FA} is the number of false rejections, and $N_{impostor}$ is the total number of false evaluations.

3 Experiments and results

In this part, our experimental processes and training configuration details were introduced, and our method was compared with other methods. Then, our model was trained on the VoxCeleb2 [28] dataset, and our methods were evaluated for the effectiveness of our framework performance on the VoxCeleb1 [29] test dataset.

3.1 Experimental environment

The parameters of experimental environment were shown in Table 1.

3.2 Experimental dataset and training details

3.2.1 Experimental dataset

In order to verify the effectiveness of our proposed framework, extensive experiments were conducted on the VoxCeleb1 and VoxCeleb2 datasets. We trained our proposed model on the development dataset of VoxCeleb2. The development dataset of VoxCeleb2 contains 1,092,009 utterances of 5994 samples. All models in the experiment were used to verify the performance of the model on the VoxCeleb1 test dataset. The VoxCeleb1 dataset contained 153,357 utterances from 1251 samples; among them, the VoxCeleb2 development dataset and the VoxCeleb1 test dataset were completely disjoint (there was no common audio signal). In addition, the VoxCeleb1 dataset provided three versions of the test dataset: VoxCeleb1 test dataset, VoxCeleb1-E test dataset, and VoxCeleb1-H test dataset. The VoxCeleb1 and VoxCeleb2 data datasets were summarized in Table 2.

3.2.2 Training details

We used Adam optimizer in our experiments, and set the initial learning rate as 10^{-3}. During training, we used a fixed length 2-s temporal segment, extracted randomly from

Table 2 VoxCeleb1 and VoxCeleb2 dataset

Dataset	# of speakers	# of utterances	# of pair
VoxCeleb2 dev	5994	1,092,009	-
VoxCeleb1 dev	1211	148,642	-
VoxCeleb1 test	40	4715	37720
VoxCeleb1-E test	1251	145,375	581,480
VoxCeleb1-H test	1190	138,137	552,536

each utterance. Spectrograms were extracted with a hamming window of width 25 ms and step 10 ms. For the Res-CASP model, the 64-dimensional log filter bank features were used as the input to the network. Mean and variance normalization (MVN) was performed by applying instance normalization to the network input. Since the VoxCeleb dataset consists mostly of continuous speech, voice activity detection (VAD) was not used in training and testing. The training time of the Res-CASP model was about 4 days, and a total of 200 epochs were trained for each experiment. In order to minimize the effect of random initialization, all experiments were repeated three times independently. The trained deep residual network model was evaluated on the VoxCeleb1 test dataset. Ten 4-s time datasets were sampled at fixed intervals from each test segment and calculated the similarity between all possible combinations (10×10 = 100) in each pair of segments. The average of 100 similarities was used as the score.

3.3 Structure of deep residual network Res-CASP

As shown in Table 3, the Res-CASP was composed of a ResNet and a CASP layer. The ResNet was used to extract higher-dimensional abstract features with optimal classification performance, which was composed of multiple Res-Blocks; the CASP layer was used to aggregate frame-level features of the ResNet. Finally, the trained model was used for final speaker recognition.

As shown in Table 3, Conv1-4 was used as the backbone of Res-CASP architecture for scale conversion and depth conversion, and the algorithm used convolutional layers to obtain abstract features of utterance. After each convolution operation, a ReLU activation function and a BN batch normalization were added to the model which had nonlinear feature conversion capabilities. The convolutional layers in residual blocks Res1-4 used 32, 64, 128, and 256 convolutional kernels of size 3 × 3, respectively, and the stride was set to 1. Conv1 used 32 convolution kernels of size 7 × 7, the stride was set to 1. Conv2-4 used 64, 128, and 256 convolution kernels of size 1 × 1, and the stride was set to 2. Therefore, frame-level features of ResNet were aggregated into utterance-level features by a CASP layer. Each signal dimension corresponded to a 64 × 200 residual network input, and 512-dimensional frame-level features were generated by deep residual model. A fully

Table 3 VoxCeleb1 and VoxCeleb2 dataset

Layer name	Kernel size	Strides	Output size
Conv1	7 ×7,32	1 ×1	(32,64,200)
Res1	$\begin{bmatrix}3\times3,32\\3\times3,32\end{bmatrix}\times 3$	1 × 1	(32,64,100)
Conv2	1 ×1,64	2 ×2	(64,32,100)
Res2	$\begin{bmatrix}3\times3,64\\3\times3,64\end{bmatrix}\times 4$	1 × 1	(64,32,100)
Conv3	1 ×1,128	2 ×2	(128,16,50)
Res3	$\begin{bmatrix}3\times3,128\\3\times3,128\end{bmatrix}\times 6$	1 × 1	(128,16,50)
Conv4	1 ×1,256	2 ×2	(256,8,25)
Res4	$\begin{bmatrix}3\times3,256\\3\times3,256\end{bmatrix}\times 3$	1 × 1	(256,8,25)
Reshape	-	-	(2048,25)
CASP	1 × 512	1 × 1	(512)
FC	-	-	(512)

connected (FC) layer was used to constrain the embedding vector to a 512-dimensional unit vectors. Finally, text-independent speaker recognition was performed by the ACLL.

3.4 Res-CASP parameters selection

The Res-CASP model was trained by text-independent speaker recognition framework.

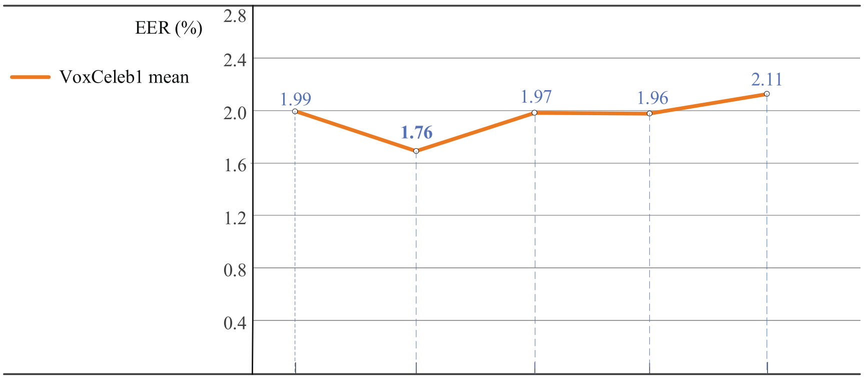
On the one hand, the training process of the Res-CASP model was a process of continuously optimizing parameters in the residual network. In order to prevent the Res-CASP model from overfitting during the learning process, the $L2$ regularization [30] mechanism was introduced into the FC layer. The Adam [31] optimizer was used in experiments, and its initial learning rate was 0.001, which was reduced by 5% every 5 epochs.

On the other hand, abstract features of the FC layer were fed into the optimized loss function at the end of each iteration of the deep residual network. In the training phase, parameters were optimized in the loss function. Because hyper-parameters of m and s in the ACLL were sensitive and fixed, relatively. In order to find the best experimental configuration for m and s, experiments were set up to explore.

On the premise of 64-dimensional log filter bank feature vectors, the hyper-parameters m was set to 0.1, 0.2, 0.3, 0.4, and 0.5, and s was fixed at 30. As shown in Fig. 5, with the increase of the hyper-parameters m, the EER of speaker recognition decreased first and then increased. Therefore, in order to have the best stability performance and the lowest EER for text-independent speaker recognitions, the best EER recognition performance was obtained when $m = 0.2$, $s = 30$ and dimensions of the log filter bank feature vectors was set to 64.

3.5 Performance analysis of speaker recognition

In order to verify the rationality of our proposed framework, two groups of experiments were designed to perform text-independent speaker recognition on the VoxCeleb2 dataset.



Loss s=30	m = 0.1	m = 0.2	m = 0.3	m = 0.4	m = 0.5
Models	Res-CASP				
Loss	ACLL				
EER	1.99	**1.76**	1.97	1.96	2.11

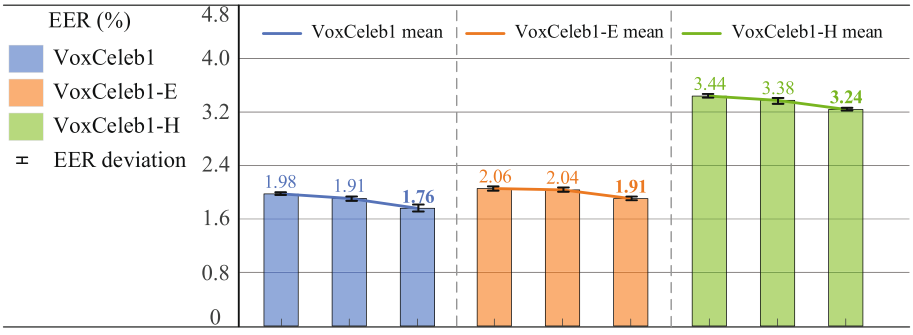
Fig. 5 The average EER of text-independent speaker recognition at $s = 30$, $m = 0.1 \sim 0.5$

In first group of experiments, the CASP was used as an aggregated frame-level features. AM-Softmax, AAM-Softmax and ACLL were used as the loss function. The text-independent speaker recognition was performed on the Res-CASP model. As shown in Fig. 6, when the CASP was used as the aggregation layer and the ACLL, AAM, and AM were used as the loss function, the average EER was 1.76%, 1.91%, and 1.98% on VoxCeleb1; 1.91%, 2.04%, and 2.06% on VoxCeleb1-E; and 3.24%, 3.38%, and 3.44% on VoxCeleb1-H, respectively. When ACLL was used as the loss function, the model Res-CASP obtained best performance for text-independent speaker recognition on three test datasets, where AM and AAM represented AM-Softmax and AAM-Softmax, respectively. The AM and AAM were margin-based loss function, and the ACLL was used to adjust the weight ratio of simple samples and difficult samples by adaptive methods. In the proposed framework, the ACLL was more effective in text-independent speaker recognition than AM and AAM, which indicated that the log filter bank signal could be effectively extracted by adaptively adjusting simple samples and difficult samples.

In the second group of experiments, the ACLL was used as the loss function; Res-TAP, Res-ASP, and Res-CASP models were used for text-independent speaker recognition. As shown in Fig. 7, the Res-TAP, the Res-ASP, and the Res-ASP achieved an average EER of 2.09%, 1.92%, and 1.76% on VoxCeleb1; 2.26%, 2.08%, and 1.91% on VoxCeleb1-E; and 3.76%, 3.59%, and 3.24% on VoxCeleb1-H, respectively. The Res-CASP achieved a better speaker recognition performance on three test datasets. In the case of the same model parameters, the Res-CASP obtained better speaker recognition performance than Res-TAP and Res-ASP, which indicated that our model could extract features information effectively. The speaker recognition performance of Res-ASP and Res-CASP were better than Res-TAP, which indicated that the attention mechanism-based aggregation layer could capture relevant information of signal features effectively.

3.6 Comparison of the results of different experimental methods

The proposed method was compared with the current recognition methods based on Res-CASP model, which were applied to the VoxCeleb1 and VoxCeleb2 dataset. As shown in



Loss	AM	AAM	ACLL	AM	AAM	ACLL	AM	AAM	ACLL
Models	**Res-CASP**								
Mean ± Standard deviation(%)	1.98 ±0.02	1.91 ±0.04	**1.76** **±0.09**	2.06 ±0.04	2.04 ±0.04	**1.91** **±0.03**	3.44 ±0.03	3.38 ±0.07	**3.24** **±0.02**

Fig. 6 Text-independent speaker recognition based on AM, AAM, and ACLL, where AM, AAM, and ACLL were used as the loss function, respectively. The EER experimental results of text-independent speaker recognition on the Res-CASP model

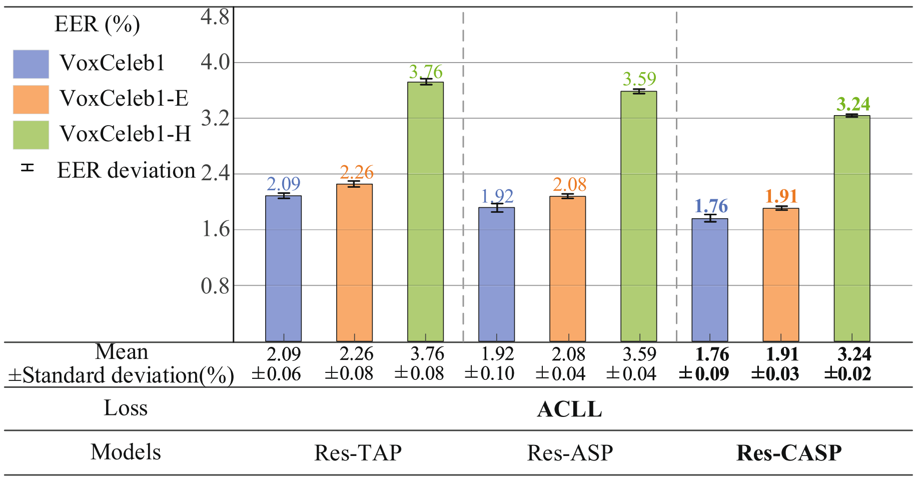


Fig. 7 Text-independent speaker recognition based on Res-TAP, Res-ASP, and Res-CASP models. The ACLL was used as the loss function, and the text-independent speaker recognition were performed on Res-TAP, Res-ASP, and Res-CASP models, respectively

Table 4, the different experimental methods of speaker recognition were carried out, and the similar methods were followed to evaluate the recognition performance. A method based on an ACLL and a deep residual network was proposed for TI-SR system in this paper.

Firstly, a ResNet and CASP aggregation layer was used to build a Res-CASP model framework. As shown in Fig. 8, ResNet and GhostVLAD aggregation layers are used to build a speaker recognition framework [11]. The experimental results showed that the proposed method was improved EER of 1.11%, 1.04%, and 1.69%, which were lower than theirs on Vox1, Vox1-E, and Vox1-H, respectively, where Vox1, Vox1-E, and Vox1-H denoted the VoxCeleb1, VoxCeleb1-E, and VoxCeleb1-H test dataset, respectively. Therefore, the CASP layer could aggregate more useful speaker features information.

Secondly, the ACLL was used as the loss function to perform text-independent speaker recognition experiments. As shown in Fig. 8. We used AAM-Softmax as the loss function [32]. The experimental results showed that the proposed method was 0.56% on Vox1, 0.88% on Vox1-E, and 1.69% lower on Vox1-H than theirs. Therefore, the ACLL could distinguish different categories feature margins.

Table 4 Comparison of text-independent speaker recognition with related methods in recent years

				EER (%)		
Studies	**Models**	**Encoding Layer**	**Loss**	**Vox1**	**Vox1-E**	**Vox1-H**
Xu et al. 2020 [34]	ResNet-50	Average Dist.	Triplet+N-pair+ Argular+softmax	3.48	-	-
Xie et al. 2019 [12]	Thin-ResNet-34	GhostVLAD	Softmax	3.22	3.13	5.06
Yu et al. 2019 [35]	ResNet-50	TAP	EAM-Softmax	2.94	-	-
Nagrani et al. 2020 [11]	Thin-ResNet-34	GhostVLAD	Softmax	2.87	2.95	4.93
Jung et al. 2019 [36]	ResNet-34	SPE	A-Softmax	2.61	-	-
Xiang et al. 2019 [32]	TDNN (x-vector)	-	AAM-Softmax	2.24	2.76	4.73
Chung et al. 2020 [23]	Fast ResNet-34	TAP	AP	2.22	-	-
Kye et al. 2020 [33]	ResNet-34	TAP	NP + Softmax	2.08	-	-
Ours	Res-CASP	CASP	ACLL	1.76	1.91	3.24

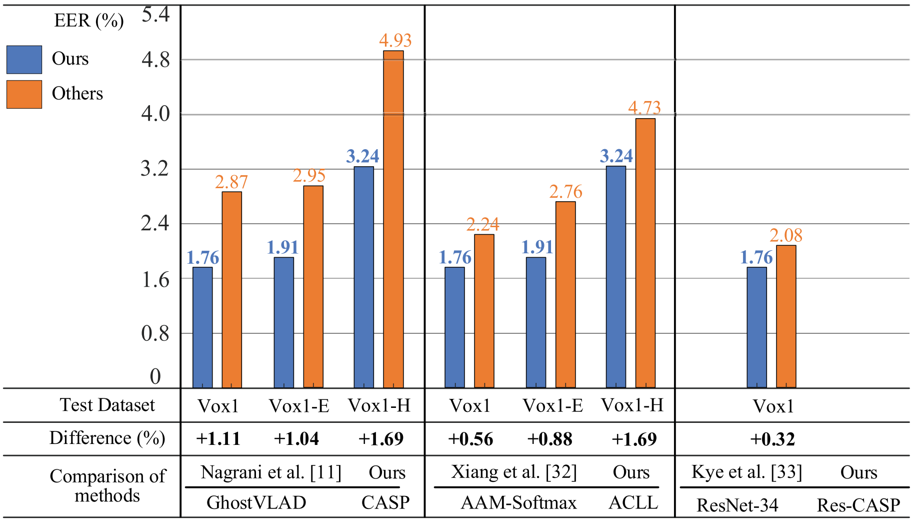


Fig. 8 Comparison with traditional methods

Thirdly, on the basic of the ResNet, we fused the CASP which captured abstract local features. And the Res-CASP were used for text-independent speaker recognition. As shown in Fig. 8, a ResNet was used to conduct text-independent speaker recognition experiments [33]. The experimental results showed that the proposed method on Vox1 test dataset was lower 0.32% lower than theirs, which indicated that the combination of ResNet and ACLL was more effective for speaker recognition. Therefore, the Res-CASP could extract more effectively information for text-independent speaker recognition.

Therefore, our method could achieve the lowest EER of text-independent speaker recognition on VoxCeleb1, VoxCeleb1-E, and VoxCeleb1-H test dataset, which was 1.76%, 1.91%, and 3.24%, respectively. Experiment verified the effectiveness of our proposed text-independent speaker recognition based on the Res-CASP. Finally, the comparison of related methods was summarized as shown in Table 4.

4 Conclusion

A method of text-independent speaker recognition based on a deep residual network Res-CASP was proposed in this paper. The CASP layer could assign different weights to each sample and could extract more useful relevant information. The proposed method was applied to the VoxCeleb2 dataset for model training, and the EER could achieve the best speaker recognition performance. In this paper, our innovations mainly included two aspects. Firstly, the Res-CASP model constructed from ResNet and CASP was proposed and used for text-independent speaker recognition. Secondly, the mining strategy of signal features was applied to the text-independent speaker recognition by using ACLL as the loss function. Compared with existing studies, our model had a better text-independent speaker recognition performance and could achieve the lowest EER recognition results on the VoxCeleb1, VoxCeleb1-E, and VoxCeleb1-H test dataset.

Abbreviations

CASP: Convolutional attention statistics pooling ; ACLL: Adaptive curriculum learning loss; EER: Equal error rate; ResNet: Residual network; SR: Speaker recognition; SV: Speaker verification; SI: Speaker identification; TD-SR: Text-dependent speaker recognition; TI-SR: Text-independent speaker recognition; TAP: Temporal averaging pooling; ASP: Attention statistical pooling; Res-CASP: Deep residual network and CASP; FC: Fully connected; Res-Blocks: Residual blocks; EMA:

Exponential moving average; FA: False acceptance; FR: False rejection; BN: Batch normalization; AM: AM-Softmax; AAM: AAM-Softmax; Vox1: Voxceleb1; Vox1-E: VoxCeleb1-E; Vox1-H: VoxCeleb1-H

Acknowledgements
Not applicable.

Authors' contributions
RD designed the framework, conducted the experiments, and wrote the manuscript. QZ carried out the experiments, analyzed the results, and presented the discussion and conclusion parts. All authors read and approved the final manuscript.

Authors' information
[1]School of Physics and Telecommunication Engineering, South China Normal University, Guangzhou 510006, China.
[2]South China Academy of Advanced Optoelectronics, South China Normal University, Guangzhou 510006, China.

Funding
This research was funded by the Natural Science Foundation of Guangdong Province (no. 2019A1515011940), in part by the Science and Technology Program of Guangzhou under (no. 2019050001, no. 202002030353), in part by the Science and Technology Planning Project of Guangdong Province under (no. 2017B030308009), in part by the Special Project for Youth Top-Notch Scholars of Guangdong Province under (no. 2016TQ03X100).

Declarations

Availability of data and materials
The database that supports the conclusions of this article is available in the [VoxCeleb [21, 22] database] repository [unique persistent identifier and hyperlink to the dataset at https://www.robots.ox.ac.uk/ vgg/data/voxceleb/ .]

Competing interests
The authors declare that they have no competing interests.

Received: 4 February 2021 Accepted: 2 July 2021
Published online: 23 July 2021

References

1. J. P. Campbell, Speaker recognition: a tutorial. Proc. IEEE. **85**(9), 1437–1462 (1997)
2. J. Hansen, T. Hasan, Speaker recognition by machines and humans: a tutorial review. IEEE Signal Proc. Mag. **32**(6), 74–99 (2015)
3. Z. Chunlei, K. Kazuhito, J. H. L. Hansen, Text-independent speaker verification based on triplet convolutional neural network embeddings. IEEE/ACM Trans. Audio Speech Lang. Process. **26**(9), 1633–1644 (2018)
4. R. Togneri, D. Pullella, An overview of speaker identification: accuracy and robustness issues. IEEE Circ. Syst. Mag. **11**(2), 23–61 (2011)
5. A. Larcher, K. A. Lee, B. Ma, H. Li, Text-dependent speaker verification: classifiers, databases and rsr2015. Speech Commun. **60**(3), 56–77 (2014)
6. J. Rohdin, A. Silnova, M. Diez, O. Plchot, P. Matějka, L. Burget, O. Glembek, End-to-end dnn based text-independent speaker recognition for long and short utterances. Comput. Speech Lang. **59**, 22–35 (2020)
7. Z. Bai, X.-L. Zhang, J. Chen, Cosine metric learning based speaker verification. Speech Commun. **118**, 10–20 (2020)
8. C. Zhang, K. Koishida, in *Interspeech 2017*, End-to-end text-independent speaker verification with triplet loss on short utterances (ISCA, 2017), pp. 1487–1491
9. H. Bredin, in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Tristounet: triplet loss for speaker turn embedding (IEEE, 2017), pp. 5430–5434. corrabs / 1609.04301
10. S. Wang, Z. Huang, Y. Qian, K. Yu, Discriminative neural embedding learning for short-duration text-independent speaker verification. IEEE/ACM Trans. Audio Speech Lang. Process. **27**(11), 1686–1696 (2019)
11. A. Nagrani, J. S. Chung, W. Xie, A. Zisserman, Voxceleb: large-scale speaker verification in the wild. Comput. Speech Lang. **60**, 101027 (2020)
12. W. Xie, A. Nagrani, J. S. Chung, A. Zisserman, in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Utterance-level aggregation for speaker recognition in the wild (IEEE, 2019), pp. 5791–5795. corrabs / 1902.10107
13. Z. Zhao, H. Duan, G. Min, Y. Wu, Z. Huang, X. Zhuang, H. Xi, M. Fu, A lighten cnn-lstm model for speaker verification on embedded devices. Futur. Gener. Comput. Syst. **100**, 751–758 (2019)
14. T. Bian, F. Chen, L. Xu, Self-attention based speaker recognition using cluster-range loss. Neurocomputing. **368**, 59–68 (2019)
15. F. Richardson, D. Reynolds, N. Dehak, Deep neural network approaches to speaker and language recognition. IEEE Sig. Process Lett. **22**(10), 1671–1675 (2015)
16. N. N. An, N. Q. Thanh, Y. Liu, Deep CNNs with self-attention for speaker identification. IEEE Access. **7**, 85327–85337 (2019)
17. H. Taherian, Z.-Q. Wang, J. Chang, D. Wang, Robust speaker recognition based on single-channel and multi-channel speech enhancement. IEEE/ACM Trans. Audio Speech Lang. Process. **28**, 1293–1302 (2020)
18. K. Okabe, T. Koshinaka, K. Shinoda, Attentive statistics pooling for deep speaker embedding. arXiv preprint arXiv:1803.10963, 2252–2256 (2018)

19. O. Boujelben, M. Bahoura, Efficient fpga-based architecture of an automatic wheeze detector using a combination of MFCC and SVM algorithms. J. Syst. Archit. **88**, 54–64 (2018)
20. A. Sithara, A. Thomas, D. Mathew, Study of MFCC and IHC feature extraction methods with probabilistic acoustic models for speaker biometric applications. Procedia Comput. Sci. **143**, 267–276 (2018)
21. K. He, X. Zhang, S. Ren, J. Sun, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Deep residual learning for image recognition (IEEE, 2016), pp. 770–778. corrabs / 1512.03385
22. R. Jahangir, W. T. Ying, N. A. Memon, G. Mujtaba, I. Ali, Text-independent speaker identification through feature fusion and deep neural network. IEEE Access. **PP**(99), 1 (2020)
23. J. S. Chung, J. Huh, S. Mun, M. Lee, H. S. Heo, S. Choe, C. Ham, S. Jung, B.-J. Lee, I. Han, In defence of metric learning for speaker recognition. arXiv preprint arXiv:2003.11982, 2977–2981 (2020)
24. F. Wang, J. Cheng, W. Liu, H. Liu, Additive margin softmax for face verification. IEEE Signal Proc. Lett. **25**(7), 926–930 (2018)
25. J. Deng, J. Guo, N. Xue, S. Zafeiriou, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Arcface: additive angular margin loss for deep face recognition (IEEE, 2019), pp. 4690–4699. corr ABS / 1801.07698
26. W. Liu, Y. Wen, Z. Yu, M. Yang, in *ICML*, Large-margin softmax loss for convolutional neural networks, vol. 2 (corrabs / 1612.02295, 2016), p. 7
27. Y. Huang, Y. Wang, Y. Tai, X. Liu, P. Shen, S. Li, J. Li, F. Huang, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Curricularface: adaptive curriculum learning loss for deep face recognition (IEEE, 2020), pp. 5901–5910. corrabs / 2004.00288
28. J. S. Chung, A. Nagrani, A. Zisserman, Voxceleb2: deep speaker recognition. arXiv preprint arXiv:1806.05622, 1086–1090 (2018)
29. A. Nagrani, J. S. Chung, A. Zisserman, Voxceleb: a large-scale speaker identification dataset. arXiv preprint arXiv:1706.08612, 2616–2620 (2017)
30. F. Li, J. M. Zurada, W. Wu, Smooth group l1/2 regularization for input layer of feedforward neural networks. Neurocomputing. **314**, 109–119 (2018). https://doi.org/10.1016/j.neucom.2018.06.046
31. D. P. Kingma, J. Ba, Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980, 1–15 (2014)
32. X. Xiang, S. Wang, H. Huang, Y. Qian, K. Yu, in *2019 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Margin matters: towards more discriminative deep neural network embeddings for speaker recognition (IEEE, 2019), pp. 1652–1656. corrabs / 1906.07317
33. S. M. Kye, Y. Jung, H. B. Lee, S. J. Hwang, H. Kim, Meta-learning for short utterance speaker recognition with imbalance length pairs. arXiv preprint arXiv:2004.02863, 1652–1656 (2020)
34. J. Xu, X. Wang, B. Feng, W. Liu, Deep multi-metric learning for text-independent speaker verification. Neurocomputing. **410**, 394–400 (2020)
35. Y.-Q. Yu, L. Fan, W.-J. Li, in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Ensemble additive margin softmax for speaker verification (IEEE, 2019), pp. 6046–6050
36. Y. Jung, Y. Kim, H. Lim, Y. Choi, H. Kim, Spatial pyramid encoding with convex length normalization for text-independent speaker verification. arXiv preprint arXiv:1906.08333, 2982–2986 (2019)

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.