Computing and Informatics, Vol. 43, 2024, 819–848, doi: 10.31577/cai_2024_4_819

MULTI-STREAM CONVOLUTIONAL NEURAL NETWORK WITH FREQUENCY SELECTION FOR ROBUST SPEAKER VERIFICATION

Wei Yao

Key Laboratory of Technology in Rural Water Management of Zhejiang Province
College of Electric Engineering
Zhejiang University of Water Resources and Electric Power
Hangzhou, China
e-mail: `yaowei@zjweu.edu.cn`

Shen Chen*

Wanbang Digital Energy Co., Ltd. (China)
Hangzhou, China
e-mail: `chenshen@next-aiot.cn`

Jiamin Cui, Yaolin Lou

Key Laboratory of Technology in Rural Water Management of Zhejiang Province
College of Electric Engineering
Zhejiang University of Water Resources and Electric Power
Hangzhou, China
e-mail: `{cuijm, louyl}@zjweu.edu.cn`

Abstract. Speaker verification aims to verify whether an input speech corresponds to the claimed speaker, and conventionally, this kind of system is deployed based on single-stream scenario, wherein the feature extractor operates in full frequency range. In this paper, we hypothesize that machine can learn enough knowledge to do classification task when listening to partial frequency range instead of full

* Corresponding author; Wei Yao and Shen Chen contributed equally to the work.

frequency range, which is so called frequency selection technique, and further propose a novel framework of multi-stream Convolutional Neural Network (CNN) with this technique for speaker verification tasks. The proposed framework accommodates diverse temporal embeddings generated from multiple streams to enhance the robustness of acoustic modeling. For the diversity of temporal embeddings, we consider feature augmentation with frequency selection, which is to manually segment the full-band of frequency into several sub-bands, and the feature extractor of each stream can select which sub-bands to use as target frequency domain. Different from conventional single-stream solution wherein each utterance would only be processed for one time, in this framework, there are multiple streams processing it in parallel. The input utterance for each stream is pre-processed by a frequency selector within specified frequency range, and post-processed by mean normalization. The normalized temporal embeddings of each stream will flow into a pooling layer to generate fused embeddings. We conduct extensive experiments on VoxCeleb dataset, and the experimental results demonstrate that multi-stream CNN significantly outperforms single-stream baseline with 20.53 % of relative improvement in minimum Decision Cost Function (minDCF) and 15.28 % of relative improvement in Equal Error Rate (EER).

Keywords: Deep learning, speaker verification, convolutional neural network, multi-stream, frequency selection

Mathematics Subject Classification 2010: 97R40

1 INTRODUCTION

Deep learning has achieved outstanding success in various speech-oriented tasks, such as auto speech recognition [1, 2], speaker recognition [3, 4, 5, 6] and speaker diarization [7, 8], etc. The deep learning paradigm is addressed to extract highly abstracted representations by means of well-designed neural networks based on the feed-in data. Most commonly, there are three scenarios to train neural network in deep learning, which are supervised learning [9], semi-supervised learning [10] and unsupervised learning (or, more precisely, self-supervised learning) [11], respectively. In addition, supervised learning with abundant labelled data is the most widely used scenario [12], which is also the scenario used in this paper.

Speaker recognition is the field of recognizing speaker identities based on their voices. In general, it can be clarified into either

1. speaker verification or
2. speaker identification.

Speaker verification aims to answer the question “is it from certain speaker?” with single utterance or “are they from the same speaker?” with pairwise utterances.

Speaker identification is used to answer the question "who is speaking?" among a set of enrolled speakers. Speaker verification is one case of biometric authentication, where user provides their biometric characteristics in form of voiceprint as passwords. The greatest challenge of the speaker verification task is the effective usage of datasets obtained from the real world under noisy and unconstrained conditions [13]. In this paper, we aim to address this challenge and propose a new framework to extract robust speaker embeddings.

1.1 Frequency Selection

Normally, the features used for training and testing are extracted in full frequency band, and they are usually low dimensional. As features play an important role in speaker verification system, if we use just partial frequency range instead of full frequency range, will the system perform equally well? Following this assumption, we first segment the full-band into several sub-bands using a frequency selector, e.g., low frequency sub-band and high frequency sub-band, and use these features to train several single-stream systems respectively, eager to witness the impact of frequency domain on system performance. The feature extractor can select which sub-bands to use as a target frequency domain to generate frame-level features, and we call this idea as frequency selection.

1.2 Resolution and Problem Statement

We hypothesize that machine could dramatically benefit from our proposed frequency selection technique. On the other hand, Convolutional Neural Network (CNN) is one kind of widely used neural networks in image recognition. More recently, CNN is introduced to speaker recognition and achieves competitive results [5, 6], compared with the most famous Time-Delay Neural Network (TDNN) and its variations [14]. Despite demonstrating encouraging outcomes, CNN for speaker verification is remaining an open topic, which requires more efforts to achieve breakthrough. By investigating various ways of doing so, we bridge frequency selection and CNN, and propose a new framework of multi-stream CNN.

However, this approach may prompt some questions: Why to use multi-stream if single-stream can offer us a high enough accuracy? Can machine learn enough knowledge to handle classification task by only listening to partial frequency range? Demonstrated by our experimental results, these are part of questions we are going to delve into and find out answer in this paper.

1.3 Contributions

Most of the speaker verification systems are deployed based on single-stream within full frequency range. To the best of our knowledge, this paper is the first to investigate the impact of frequency domain and to improve system performance by means of frequency selection. Our contributions are as follows:

1. We explore the performance of neural network by feeding in "partial" features extracted from speech within sub-bands of frequency instead of conventionally used full-band. And we find that machine can perform equivalently well in some sub-bands, which are also beneficial to improve the performance of multi-stream system.
2. We propose the idea of frequency selection, and a novel framework of multi-stream CNN based on it for speaker verification.
3. We make our work open source, and it is available to download at `https://github.com/ShaneRun/multistream-CNN`.

1.4 Structure of This Paper

This paper is organized as follows. We review the related work in Section 2. Section 3 describes our proposed method in detail. Section 4 presents experiments for pairwise verification, which consists of dataset, training and results. We also make a comprehensive comparison in this section to demonstrate the efficacy and understand the influence of the frequency selection. Section 5 contains discussion and future work. Section 6 is the conclusion of our work.

2 RELATED WORK

Most of the works done so far on speaker verification are based on a single-stream framework as illustrated in Figure 1. In general, it is comprised of train process (including validation) and test process, and can be classified into several modules, including front-end, encoder, back-end, loss and similarity/score. The train process is to tune network parameters of the encoder using abundant labelled data. After training, the back-end and loss module are not used any longer, whereas the shared block (including front-end and encoder), which is enclosed by imaginary line, will be inherited by the test process. The test process is to make a decision on whether or not the utterance pair is from the same speaker.

The front-end module, or rather the feature extractor, is used to extract quality frame-level features for subsequent signal processing by converting acoustic waveform into a relatively lower dimensional representations, such as the well known Mel Filter Bank Energies (MFBE) [15, 16] and Mel Frequency Cepstral Coefficients (MFCC) [17, 18]. The encoder is used to extract unique speaker embeddings in utterance-level through the deep neural network, which is known as identity vector, e.g., "i-vector" [19], "x-vector" [9], and "r-vector" [20]. The back-end module is for the post-processing of speaker embeddings in order to enlarge inter-class distance and reduce intra-class distance. The most popular approaches for back-end processing contain the Gaussian back-end model [21], Probabilistic Linear Discriminant Analysis (PLDA) [22], and neural-based model [23]. The loss module is used only in the training phase and plays an important role because it is

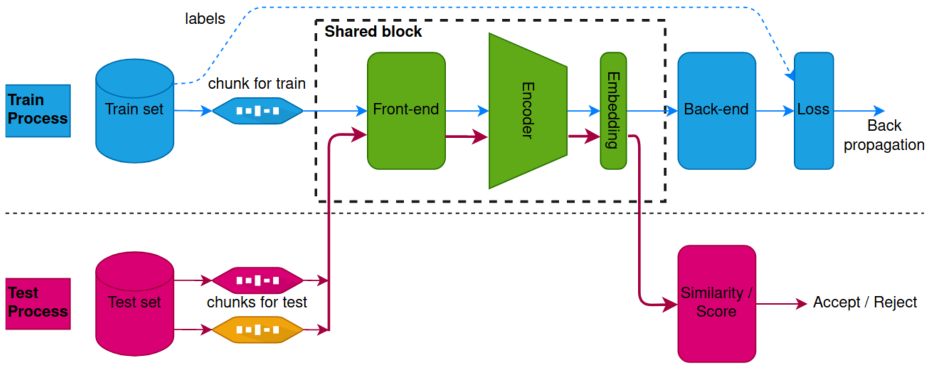


Figure 1. A schematic illustration of a speaker verification system consisting of the train process and test process. The train process is to tune the parameters of the encoder in order to minimize the loss using back propagation. The test process is to decide on whether to accept or reject under the tested pair of utterances from the test set.

actually defining the goal function. In [6], extensive evaluations of the most popular loss function, including softmax loss, angular softmax loss, triplet loss and angular prototypical loss, etc., are presented, and it is also demonstrated that metric learning objectives outperform classification-based losses. The similarity module generates probability or score based on the embeddings pair by adopting Euclidean distance or Cosine similarity, and then decides on the accept or reject.

Over the years, multi-stream approach has attracted a lot of attentions in the deep learning field, such as Computer Vision (CV) [24, 25, 26] and Automatic Speech Recognition (ASR) [27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37]. [24] and [25] both propose a multi-stream CNN architecture to recognize human actions and gestures. [24] is implemented by combining novel human-related streams containing one appearance and one motion stream with the traditional streams. Whereas [25] decomposes the original image into several equal-sized streams and learn representations by a CNN for each stream. Then the features learned from all streams are fused into a unified feature map, which is subsequently fed into a neural network to recognize gestures. In [26], a multi-stream framework, which is comprised of motion stream, spatial stream and structural stream, is designed for unmanned aerial vehicles video aesthetic quality assessment.

In [27], a novel effort to estimate word error rate uses a multi-stream end-to-end architecture based on a combination of four independent streams which deal with decoder, acoustics, textual and phonotactics features in parallel. [28, 29, 30, 31] all employ several parallel streams by using audio-visual strategy. The reason behind these approaches is to address the problem of speech recognition by leveraging visual information to improve the performance of ASR. Nevertheless, [32, 33, 34, 35, 36, 37] aim to capture diverse information from audio only for end-to-end ASR. For instance,

[32] presents a framework based on joint attention with multiple audio streams in parallel. More practically, to address the problem of massive computation and memory requirements during training resulting from increasing number of streams, [33] introduces a two-stage training scenario for end-to-end ASR, where the training of the feature extractor and the attention fusion module is processed in separated stages.

It can be perceived that multi-stream framework is successfully used in ASR inspired by observing multiple streams in parallel. However, there is still limited research on speaker verification tasks. To the best of our knowledge, [34] is the most related work done so far on the speaker verification task, but in this work, the multiple streams are generated after feature extraction based on the trained intelligibility likelihood model. In addition, it only uses multi-stream in the test process. On the contrary, in our work, we first form streams by selecting the original speech signal in the frequency domain, and then design our framework to make use of multiple speech streams in both training and test processes.

3 PROPOSED METHOD

In this section, we propose a multi-stream framework using three streams processing in parallel for the speaker verification task. We present design details of all modules containing frequency selection, acoustic features, ResNet-34 based encoder, loss function and similarity.

3.1 Diagram of Framework

The proposed framework of multi-stream CNN is illustrated in Figure 2. Frequency full-band (FB) is segmented into two sub-bands after frequency selector, including Low Frequency (LF) sub-band and High Frequency (HF) sub-band. The same speech signal is processed by different streams in parallel, namely FB-stream, LF-stream, and HF-stream, respectively. In our proposed framework, FB-stream serves as a general encoder because it listens to the full frequency band, whereas LF-stream and HF-stream encoders serve as specialized encoders because they focus on listening to LF-band and HF-band of input speech. Even though the overall performance of LF-stream or HF-stream is inferior to FB-stream (this is also demonstrated by our experiments in the next section), they can still contribute to the performance of the proposed multi-stream system, mainly because they are better at extracting partial characteristics, so that the fused speaker embeddings can be more robust.

The shared block of each stream has the same structure but different in parameters. After temporal embeddings are extracted by all streams, they are then mean-normalized, and subsequently fed into a pooling layer to generate fused or final embeddings.

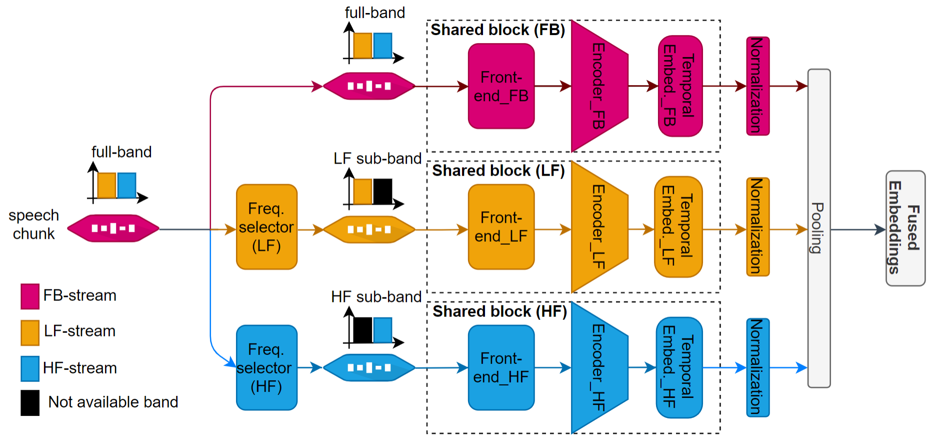


Figure 2. Proposed framework of multi-stream CNN. Shared block (FB), Shared block (LF) and Shared block (HF) have the same network structure.

Similar to conventional acoustic modeling, each stream encodes the acoustic features into highly abstracted temporal embeddings as formulated in Equation (1):

$$\vec{x}^{(s)} = Encoder^{(s)}(\vec{u}), s \in \{1, 2, 3\}, \tag{1}$$

where superscript $s \in \{1, 2, 3\}$ is denoted as index for each encoder of corresponding stream s (with 1, 2 and 3, for FB-stream, LF-stream and HF-stream, respectively), $\vec{u}$ is the input vector of chunk which is usually extracted with fixed length (2–4 seconds) randomly from speech utterances (note that each stream use the same $\vec{u}$), $\vec{x}$ is the output vector of encoder which is denoted as temporal embeddings in this article.

Then the fused embeddings in terms of $\vec{x_f}$ can be formulated as:

$$\vec{x_f} = \sum_{s=1}^{3} k_t^{(s)} \vec{x}^{(s)}, \tag{2}$$

where $k_t^{(s)}$ is the fusion weight for each temporal embeddings, and the definition of s is the same as in Equation (1).

3.2 Frequency Selection

Normally, the acoustic features are obtained with a full-band of frequency. In this article, we want to witness the machine's capability to do speaker verification with different sub-bands of frequency. We use a frequency selector on top of each stream, and by doing so, the upper and lower limits of different sub-bands can be configurable.

More specifically, the boundary between LF and HF sub-band is initially set to 1 000 Hz. The reason behind this setting is that the fundamental frequency of the complex speech tone which is also known as the pitch or f_0, lies below 500 Hz even though it differs from person to person, whereas this range might extend approximately to 1 000 Hz in some cases [38]. The overview of our frequency segmentation is shown in Figure 3.

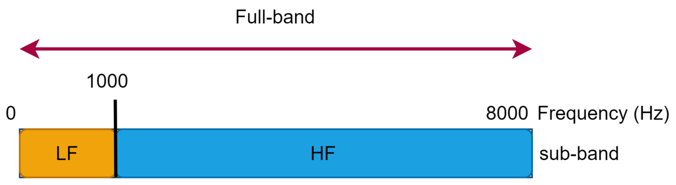


Figure 3. Frequency segmentation

We implement frequency selection in PyTorch [39] by using a class called "torchaudio.transforms.MelSpectrogram", which is designed to create MelSpectrogram for a raw audio signal. In practice, we can adjust minimum frequency (f_{min}, default value is 0) and maximum frequency (f_{max}, default value is half of the sampling rate) in our training script so that frequency selection will be employed.

3.3 Acoustic Features

The most common forms of acoustic features are Mel Filter Bank Energies (MFBE) and Mel Frequency Cepstral Coefficients (MFCC). The procedures for computing of MFBE and MFCC features are similar, where in both cases the speech signal is first pre-processed by a pre-emphasis filter; then it is segmented into frames with overlapping and all frames are applied with a window function (normally Hamming window) in order to reduce spectrum leakage; afterwards, each frame goes through a Fourier transform to calculate filter bank energies (or power spectrum). To obtain MFCC, a Discrete Cosine Transform (DCT) is applied to the filter bank energies in order to retain the resulting coefficients, whereas the other coefficients will be discarded [40]. The reason for discarding the other coefficients is that they represent fast changes in the filter bank coefficients and these fine details seldom contribute to the system performance.

Despite the huge contribution of MFCC to speech-related tasks, there is something wrong with it as mentioned above: it requires an extra step called DCT to decorrelate coefficients of filter bank energies. Since DCT is an additional linear transform, some information in highly non-linear speech signals will be discarded undesirably. For this reason, MFBE is becoming increasingly popular due to rapid growth of end-to-end techniques using deep learning, because it can provide more information than MFCC for neural networks to delve into [15, 16, 41].

We use 40-dimensional MFBE [42] as acoustic features, with a Hamming window of 25 ms width and 10 ms step both for traning and testing. We use mel scale fbanks function in the torchaudio recipe [43]. The triangular filters are placed non-linearly on the mel scale, as depicted in Figure 4. The chunk length is extracted randomly from each utterance, and is fixed as 2 seconds and 4 seconds for training and testing, respectively.

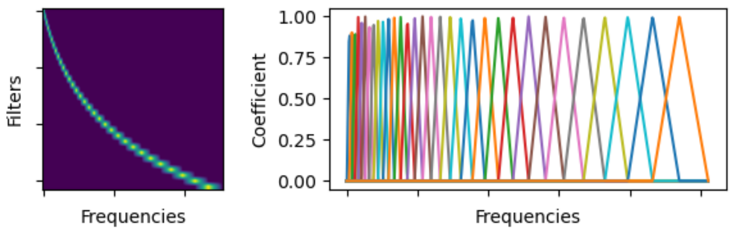


Figure 4. Mel filters configuration of Torchaudio [43]

3.4 ResNet-34 Based Encoder

ResNet, short for the Residual Network, is a form of CNN introduced in [44], and achieves extreme success in the field of image recognition. The basic idea of ResNet is to alleviate the problem of gradient vanish or explosion in training very deep neural networks by using residual blocks with skip connections. The skip connections behave as shortcut paths for gradient to flow through alternately, and allow the higher layer to learn the identity functions directly so that it can perform not worse than the lower layer.

For the last few years, some work has been done to introduce ResNet to the field of speaker recognition, and achieves encouraging results [5, 6], compared with TDNN and its variations [14]. We assume speaker recognition is somewhat similar to the image recognition, and therefore ResNet, which is very popular in image recognition, might also be widely used in this field.

We use ResNet-34 [44] with 34 hidden layers as the network structure of the encoder, as shown in Table 1. The total frames are 200 for each chunk, therefore, the size of the input feature is 40×200. The network consists of 34 convolutional layers with batch normalization and Rectified Linear Units (ReLU) activation function applying to each of them, and these layers can be grouped into Conv1, Res1, Res2, Res3, Res4 and Flatten, respectively. The output of the encoder is 512-dimensional speaker embeddings.

Layer Group	Kernel Size	Stride[a]	Output Size
Input	–	–	$40 \times 200 \times 1$
Conv1	$3 \times 3 \times 16$	1×1	$40 \times 200 \times 16$
Res1	$\begin{bmatrix} 3 \times 3 \times 16 \\ 3 \times 3 \times 16 \end{bmatrix} \times 3$	1×1	$40 \times 200 \times 16$
Res2	$\begin{bmatrix} 3 \times 3 \times 32 \\ 3 \times 3 \times 32 \end{bmatrix} \times 4$	2×2	$20 \times 100 \times 32$
Res3	$\begin{bmatrix} 3 \times 3 \times 64 \\ 3 \times 3 \times 64 \end{bmatrix} \times 6$	2×2	$10 \times 50 \times 64$
Res4	$\begin{bmatrix} 3 \times 3 \times 128 \\ 3 \times 3 \times 128 \end{bmatrix} \times 3$	2×2	$5 \times 25 \times 128$
Flatten	–	–	$5 \times 2\,048$
ASP	–	–	$4\,096$
Linear	512	–	512

[a]For stride that is not 1, it is only performed on the top layer of each residual block for down sampling, whereas the stride inside residual block is always 1.

Table 1. Network structure of ResNet-34 based encoder. ASP: Attentive Statistics Pooling.

3.5 Loss Function

The los function plays an important role in training neural networks because it estimates the error for the current state, which is then used to update the weights of the model through gradient descent and back propagation. By doing it repeatedly for massive times, the overall loss of the model tends to be minimized which is usually expected to be the global minimum.

Softmax Loss. The Softmax loss is one typical form of the loss for multi-class classification tasks, and it can accept many inputs and calculate probability for each one. The Softmax loss (in terms of L_{sm}) consists of a Softmax function followed by a cross-entropy loss, which is formulated as Equation (3):

$$L_{sm} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{\vec{W}_{y_i}^T \vec{x_i} + \vec{b}_{y_i}}}{\sum_{j=1}^{C} e^{\vec{W}_j^T \vec{x_i} + \vec{b}_j}}, \tag{3}$$

where $\vec{W}$ and $\vec{b}$ are the weight and bias vector of last layer of encoder, respectively. C is the total amount of classes (or speakers), and N is the number of utterances of each mini-batch from different speakers with embeddings $\vec{x_i}$

(extracted by encoder as defined in Equation (1)) and its corresponding label y_i.

Angular Prototypical Loss. Similar to the original prototypical loss, the angular prototypical loss uses the same batch formation, whereas the similarity metric S is changed from distance-based to cosine-based as shown in Equation (4):

$$\vec{S}_{i,k} = \omega \times \cos(\vec{x}_{i,k}, \vec{c}_k) + b, \tag{4}$$

where ω and b are learnable weight and bias, $\vec{c}_k$ is the centroid (or prototype) as shown in Equation (5):

$$\vec{c}_k = \frac{1}{M-1} \sum_{m=1}^{M-1} \vec{x}_{k,m}, \tag{5}$$

where M is the utterances number for every speaker inside mini-batch.

The angular prototypical loss is then derived as:

$$L_{ap} = -\frac{1}{N} \sum_{i=1}^{N} \log \frac{e^{\vec{S}_{i,i}}}{\sum_{k=1}^{N} e^{\vec{S}_{i,k}}}. \tag{6}$$

Softmax + Angular Prototypical Loss. The ultimate goal of designing loss function is to enlarge inter-class distance and meanwhile reduce inter-class distance. To this end, we propose a combined form of Softmax loss L_{sm} and Angular Prototypical loss L_{ap} as fused loss function of our work, which is illustrated in Equation (7):

$$L = L_{sm} + L_{ap}. \tag{7}$$

3.6 Similarity

Similarity is the scoring module used to compute the score based on the pair of speaker embeddings. The score is used subsequently to decide on accept or reject. Since open-set speaker recognition is essentially a metric learning problem, the Euclidean distance [6] is more preferred as metric of similarity than the cosine similarity (measures the cosine of angle between two high-dimensional vectors) [45].

The Euclidean distance between two speaker embeddings is formulated as:

$$\text{distance} = \parallel (\vec{x_{f_1}} - \vec{x_{f_2}}) \parallel_2 = \sqrt{\sum_{d=1}^{D} (x_{f1,d} - x_{f2,d})^2}, \tag{8}$$

where $\vec{x_{f_1}}$ and $\vec{x_{f_2}}$ are the fused embeddings of each speaker inside pair, D is the dimension of embeddings which is designed to be 512 in this article.

4 EXPERIMENTS

In this section, we present the implementation details of our proposed framework.

1. Firstly, we introduce the VoxCeleb dataset used for training, validation and testing.
2. Secondly, we present our training details and propose a practical training strategy inspired by [33].
3. Thirdly, we introduce the evaluation metrics used in our work.
4. And finally we describe our experimental results.

4.1 Dataset

Speaker verification faces many challenges in real situations related to ambient noise and short speech frame availability. Therefore, using a dataset generated from the real world for experiments is more meaningful. For this reason, we select VoxCeleb dataset for training and testing in our work. The VoxCeleb dataset is obtained from the real world, in which the speakers span a wide range of different ethnicities, accents, professions and ages. The speech of this dataset is shot in a large number of challenging auditory environments. Most crucially, all speech is degraded with the real world noise, consisting of background chatter, laughter, overlapping speech, and room acoustics, and there is a range in the quality of recording equipment and channel noise [46]. Moreover, the utterance length of the VoxCeleb dataset is distributed randomly from 4 seconds to 20 seconds.

The VoxCeleb dataset is released by the VGG group of Oxford University in two stages, as VoxCeleb1 [47] which contains 153 516 utterances (352 hours in total) from 1 251 celebrities, and VoxCeleb2 [48] which contains 1 128 246 utterances (2 442 hours in total) from 6 112 celebrities. Moreover, there is a challenge organized annually based on this dataset in order to witness how well current methods can recognize speakers from speech obtained "in the wild" since 2019 [13, 49].

The datasets used in our work are listed in Table 2. We use "VoxCeleb2-Dev" for training without data augmentation, and cleaned version of "VoxCeleb1-Test" (also referred to as "VoxCeleb1-O") [46] for validation and testing, which contains 37 720 testing pairs.

4.2 Training

Our work is implemented in PyTorch framework with details showed in Table 3. All encoders are trained in a single GPU platform with 11 GB memory for maximum 100 epochs. In order to reduce class imbalance, we apply random sampling with a maximum value of 100 utterances for each speaker in the training set. Additionally, we use the largest batch size with 400 that fits on a GPU, and the training for one stream takes approximately three days.

Stage	Dataset	# of speakers	# of utterances
Training	VoxCeleb2-Dev[a]	5 994	1 092 009
Validation	VoxCeleb1-Test[b]	40	4 874
Testing	VoxCeleb1-Test	40	4 874

[a]Development set of VoxCeleb2, which has no overlap with the identities in the VoxCeleb1. [b]Test set of VoxCeleb1, cleaned version [46], the amounts of testing pairs is 37 720.

Table 2. Dataset for training, validation and testing

Item	Value
Deep learning framework	PyTorch v1.5.1
GPU	GeForce GTX 1080 TI (single)
Optimizer	Adam
Batch size	400
Maximum epochs	100
Initial learning rate	0.001
Learning rate decay	0.95 per 10 epochs

Table 3. Training details overview

As the streams increased, the conventional training approach where all encoders are trained in parallel, is hard to implement due to massive computation and memory requirements. In order to address this problem, we adopt a more practical training approach inspired by [33], which consists of "Stage 1: Sequential training of each stream" and "Stage 2: Searching for optimal fusion weight".

Stage 1: Sequential training of each stream. In this stage, each stream will be trained in sequential mode. Before training different streams, we only need to regulate the frequency range which is designed in Figure 3. In this way, training of a large network becomes much more simple: to repeat the training of a relatively smaller network multiple times. After sequential training, the well-trained encoders will be deployed to the multi-stream system, as shown in Figure 2.

Stage 2: Searching for optimal fusion weight. Based on the scores output of each stream, we propose a simplified algorithm to address the problem of searching for optimal fusion weight. As shown in Algorithm 1, the scores output of each stream is normalized in advance using t-Distributed Stochastic Neighbor Embedding (t-SNE) approach [50] and then used as input for this algorithm. $k_t^{(1)}$ is gradually reduced with each step. For every step of $k_t^{(1)}$, $k_t^{(2)}$ is also gradually reduced with step, and repeat calculating minDCF until $k_t^{(2)}$ is smaller than the minimum weight K_{min}. The return value will be used to update the local optimum, which is defined as optimal weight under given $k_t^{(1)}$. Local optimum will be used for updating global optimum for every step of $k_t^{(1)}$ repetitively until

$k_t^{(1)}$ is smaller than minimum weight K_{min}. In our experiment, step and K_{min} are set as 0.01 and 0, respectively.

Algorithm 1 Searching for Optimal Fusion Weight

```
Require: t-SNE normalized scores $\vec{scores}$ of each stream $s \in \{1, 2, 3\}$
Ensure: optimal fusion weight $\left\{k_t^{(1)}, k_t^{(2)}, k_t^{(3)}\right\}$
  $\vec{k_t} \leftarrow [1, 0, 0]$
  repeat
    repeat
      $k_t^{(3)} \leftarrow \left(1.0 - k_t^{(1)} - k_t^{(2)}\right)$
      calculate minDCF based on $\vec{\text{scores}}$ and $\vec{k_t}$
      update local optimum
      $k_t^{(2)} \leftarrow \left(k_t^{(1)} - \text{step}\right)$
    until $k_t^{(2)} \lneqq K_{min}$
    update global optimum
    $k_t^{(1)} \leftarrow \left(k_t^{(1)} - \text{step}\right)$
  until $k_t^{(1)} \lneqq K_{min}$
```

4.3 Evaluation Metrics

For a speaker verification system with pairwise input, it is naturally a binary classifier, and the evaluation metrics are described in this section. The decision result distribution is illustrated in Figure 5, which is comprised of True Positive (TP), True Negative (TN), False Positive (FP) and False Negative (FN). The first term (True or False) is the result of prediction. If the prediction fits the ground truth, then the result is True, otherwise, the result will be False. The second term (Positive or Negative) is the category of ground truth. If the utterance pair is from the same speaker, then the ground truth is Positive, otherwise it will be Negative. Based on Figure 5, we introduce two common kinds of evaluation metrics, which are Equal Error Rate (EER) and minimum Decision Cost Function (minDCF).

EER. This is a widely used metric to determine the threshold value when false acceptance rate E_{FA} and false rejection rate E_{FR} are equal.

$$E_{FA} = \frac{N_{FN}}{N_{FN} + N_{TP}}, \tag{9}$$

$$E_{FR} = \frac{N_{FP}}{N_{FP} + N_{TN}}, \tag{10}$$

where N_{FN}, N_{TP}, N_{FP} and N_{TN} are the number of False Negative, True Positive, False Positive and True Negative, respectively.

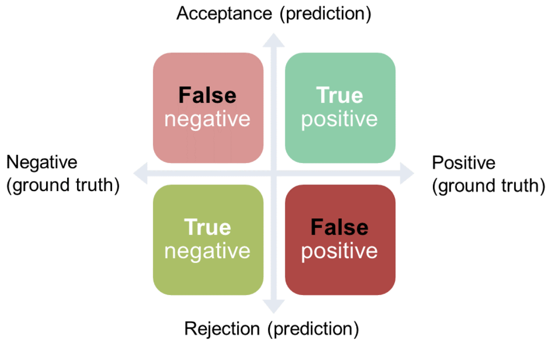


Figure 5. Distribution map of decision result. Ground truth is the label, and prediction is the score generated by the classifier.

minDCF. National Institute of Standards and Technology (NIST) has defined another metric called Decision Cost Function (DCF) in order to compare different systems at an interesting operating point [51]. DCF is designed to evaluate the overall cost on making decision errors of both missed detection and false alarm, and has served as a criterion in every NIST Speaker Recognition Challenge with some parameter adjustments in its definition [52].

$$DCF = C_{FR} P_{Target} P_{FR} + C_{FA}(1 - P_{Target}) P_{FA}, \tag{11}$$

where C_{FR} (C_{Miss} in [51]) and P_{FR} are cost and probability of missed detection; P_{Target} is a priori probability of the specified target speaker; C_{FA} ($C_{FalseAlarm}$ in [51])and P_{FA} are cost and probability of spurious detection.

In our experiment, C_{FR}, C_{FA} are set as 1, and P_{Target} is set as 0.05.

4.4 Results

We conduct several experiments to check the efficacy our proposed framework. Firstly, we build the state-of-the-art baseline as shown in Table 1 and then compare the performance of our baseline system with other existing works using both minDCF and EER metrics. Secondly, we describe our experimental results on frequency selection by adjusting the frequency range and training our baseline system repetitively for several times. Moreover, performance improvement of our proposed multi-stream framework is also illustrated. Finally, we choose some typically used feature dimension reported in literatures and repeat our experiments as done in 40-d feature dimension to see the generalized improvement level of our proposed method.

Our baseline system. As shown in Table 4, we compare our baseline system with I-Vectors [47, 53], X-Vectors [9], VoxCeleb1's approach [47] and VoxCeleb2's approach [48].

The term I-Vectors was coined around 2009/2010, where the “I” stood for “Identity” [52], and became widely used in speaker verification from then on. This approach uses one component to model variability of both speaker and channel, and extract sole low-dimensional representations of utterances. Since it is not a neural network based encoder, network type and loss function are not applicable in Table 4.

System	Train Set	Feature	Encoder	Dim.	minDCF	EER (%)
I-Vectors [47, 53][a]	Vox1	super vector[b]	–	400	0.73	8.8
X-Vectors [9]	Vox1	24-d MFBE	TDNN	512	0.393[c]	4.16
J. S. Chung et al. [47]	Vox1	spectro-gram	VGG-M CNN	1 024	0.75	10.2
J. S. Chung et al. [47]	Vox1	spectro-gram	VGG-M CNN	256	0.71	7.8
A. Nagrani et al. [48]	Vox2	spectro-gram	ResNet-34	512	0.549	4.83
A. Nagrani et al. [48]	Vox2	spectro-gram	ResNet-50	512	0.429	3.95
Xie et al. [4]	Vox2	spectro-gram	Thin-ResNet-34	512	0.35	3.22
Y. Jung et al. [54]	Vox2	64-d MFBE	ResNet-34	512	0.245	2.61
S. M. Kye et al. [55]	Vox2	40-d MFBE	ResNet-34	512	0.234	2.08
Our baseline	Vox2	40-d MFBE	ResNet-34[d]	512	**0.210**	**2.73**

[a]Work is done in [47] using method of [53]. [b]A supervector is composed by stacking the mean vectors from a Gaussian Mixture Model (GMM). [c]P_{Target} is 0.01. [d]The depth of our encoder is one quarter of original ResNet-34. [e]AP: Angular Prototype.

Table 4. Comparison of speaker verification results on VoxCeleb1-Test

In [9], concept of X-Vectors was first proposed by using deep neural network to capture speaker characteristics, and it quickly became the dominating approach for speaker recognition. The neural network used in X-Vectors is also called as Time-Delay Neural Network (TDNN), which is naturally a variation of 1-D CNN. Additionally, X-Vectors is usually post-processed by PLDA back-end for satisfying results.

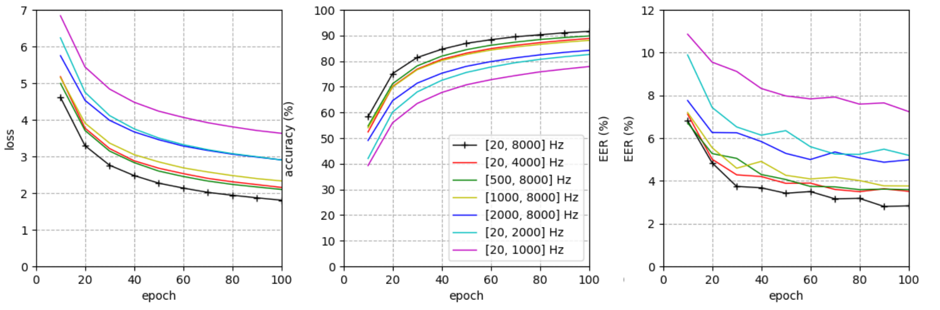
In [47], VoxCeleb1's approach based on VGG-M CNN is proposed. Different from X-Vectors and our baseline system, spectrograms is used as feature, and contrastive loss is designed as loss function.

In [48], VoxCeleb2's approach based on ResNet is proposed by using VoxCeleb2 as train set. Similar to our baseline system, speaker verification is treated as a case of metric learning, therefore Euclidean distance is applied as criteria of similarity. Both ResNet-34 and ResNet-50 are studied, and the performance of ResNet-50 is better than ResNet-34.

Details of our baseline system is depicted in Section 3. One notable thing is that almost all reference works have no description on the frequency range, as they might all use full frequency range. This is common knowledge, which is also the default setting of our baseline system. However, this common knowledge might also block us from discovering more useful and interesting things, and that is why we explore the technique of frequency selection and multi-stream.

Frequency selection and multi-stream. The learning curves of our baseline encoder and other six encoders within different frequency range are illustrated in Figure 6. The curves contain three plots for each encoder: Figure 6 a) training loss (the smaller, the better), Figure 6 b) top-1 accuracy (the higher, the better), and Figure 6 c) validation EER (the smaller, the better).

We design three sub-bands for both LF and HF sub-bands, which are LF1 ([20, 1 000] Hz), LF2 ([20, 2 000] Hz), LF3 ([20, 4 000] Hz), HF1 ([2 000, 8 000] Hz), HF2 ([1 000, 8 000] Hz) and HF3 ([500, 8 000] Hz), respectively. It can be perceived from the curves that the baseline encoder within full frequency range has the best performance, whereas LF1 encoder has the worst performance. What is more, we can further conclude that the wider the frequency range, the better the system performance will be for single-stream system.



a) Training loss b) Top-1 accuracy c) Validation EER

Figure 6. Learning curves

The evaluation results of single-stream with different frequency range and multi-stream with different combinations of single streams are depicted in Table 5. It can be perceived that the combination of FB-stream, LF2-stream and HF2-stream has the best performance. In addition, it is quite interesting that the combination of top-3 (FB-stream, LF3-stream, HF3-stream) is not the best choice, and the reason behind this phenomena might be that only the proper frequency range instead of wider frequency range can offer relatively the most benefits. Moreover, LF sub-band and HF sub-band, which is initially designed in Figure 3, is better to be adjusted from non-overlapping to overlapping. We further conduct more experiments under different dimension of input feature based on adjusted frequency segmentation, where LF and HF sub-band are designed to be $[20, 2\,000]$ Hz and $[1\,000, 8\,000]$ Hz respectively.

Stream Info	Frequency Range (Hz)	minDCF	EER (%)	Optimal Weight[a]
FB	$[20, 8\,000]$	0.2095	2.7274	–
LF1	$[20, 1\,000]$	0.4784	6.9301	–
LF2	$[20, 2\,000]$	0.3622	4.9463	–
LF3	$[20, 4\,000]$	0.2511	3.3454	–
HF1	$[2\,000, 8\,000]$	0.3420	4.7920	–
HF2	$[1\,000, 8\,000]$	0.2699	3.6326	–
HF3	$[500, 8\,000]$	0.2609	3.4677	–
FB + LF1 + HF2	$[20, 8\,000]$	0.1694	2.331	[0.37, 0.35, 0.28]
FB + LF2 + HF1	$[20, 8\,000]$	0.1667	2.356	[0.40, 0.40, 0.20]
FB + LF2 + HF2	$[20, 8\,000]$	**0.1665**	**2.297**	[0.39, 0.35, 0.26]
FB + LF2 + HF3	$[20, 8\,000]$	0.1717	2.339	[0.39, 0.35, 0.26]
FB + LF3 + HF2	$[20, 8\,000]$	0.1695	2.313	[0.37, 0.33, 0.30]
FB + LF3 + HF3	$[20, 8\,000]$	0.1737	2.383	[0.48, 0.26, 0.26]

[a]From the left to the right, the weights are for FB-stream, LF-stream, and HF-stream, respectively.

Table 5. Evaluation results of single-stream and multiple-stream

The Detection Error Tradeoff (DET) curves of FB-stream, LF-stream ($[20, 2\,000]$ Hz), HF-stream ($[1\,000, 8\,000]$ Hz) and Multi-stream are shown with red dash line, yellow dash line, blue dash line and green solid line, respectively, in Figure 7. It can be perceived that our proposed multi-stream framework has a comprehensive improvement compared with our baseline system.

Improvement vs. feature dimension. We then conduct experiments by regulating the feature dimension to 32-d and 80-d, looking forward to check the efficacy of our proposed method in other dimension. The experimental results are illustrated in Table 6, apart from feature dimension and batch size (the larger the feature dimension, the more memory requirement), all other configuration are the same for both training and testing. The batch size for training under feature dimension with 32-d and 80-d are 480 and 200, respectively.

It can be found from Table 6 that there are significant improvements in both minDCF and EER metrics from the low-dimensional to high-dimensional feature.

One thing we need to point out is that if the dimension of full-band input data is n, then the dimension of the multi-stream data will be $3 \times n$ as there are three

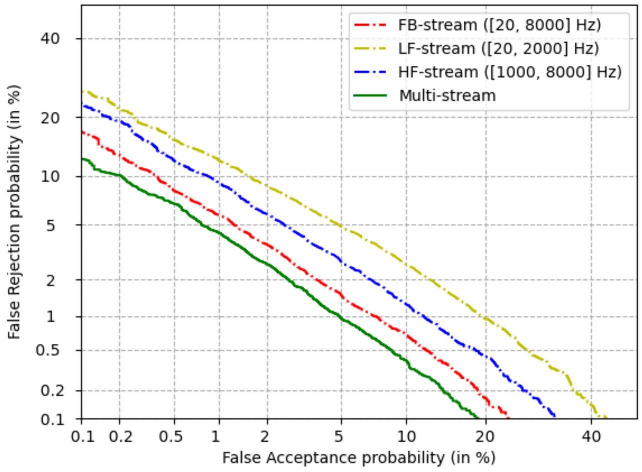


Figure 7. Curves of Detection Error Tradeoff

Feature Dim.	Stream Info [a]	minDCF	EER (%)	Optimal Weight[b]
32-d	FB-stream	0.2316	3.077	–
	LF-stream	0.4108	5.346	–
	HF-stream	0.2767	3.899	–
	Multi-stream	0.1805	2.489	$[0.36, 0.31, 0.33]$
40-d	FB-stream	0.2095	2.727	–
	LF-stream	0.3622	4.946	–
	HF-stream	0.2699	3.633	–
	Multi-stream	0.1665	2.297	$[0.39, 0.35, 0.26]$
80-d	FB-stream	0.1780	2.297	–
	LF-stream	0.3272	4.438	–
	HF-stream	0.2380	3.090	–
	Multi-stream	0.1367	1.946	$[0.36, 0.32, 0.32]$

[a]FB-stream: $[20, 8\,000]$ Hz, LF-stream: $[20, 2\,000]$ Hz, HF-stream: $[1\,000, 8\,000]$ Hz. [b]From the left to the right, the weights are for FB-stream, LF-stream, and HF-stream, respectively.

Table 6. Evaluation results on different feature dimension

encoders with different input data. This fact should be taken into account when evaluating the results of the system.

Moreover, it can be seen from Table 6, that simple increasing the dimension of the input representative vector from 40 to 80 in the full-band system gives the same improvement in EER as the proposed multi-stream approach with the $3 \times 40 = 120$ dimension. So the simple increasing the dimension seems to be more effective than the proposed approach. For this reason, we did more experiments to compare the performance between the multi-stream system and the full band system under the same feature dimension. We tried several full band systems with the same dimension of representative vector as the sum of the input data of the three ensemble subsystems. The experimental results are illustrated in Table 7. It can be seen from Table 7, that simple increasing the dimension of the input representative vector from 96 to 240 in the full-band system gives an improvement in both minDCF and EER. The full band system with dimension 96 even outperforms the multi-stream system with dimension 3×32 in both minDCF and EER. However, when dimension increases to 120, multi-stream system outperforms full band system in minDCF (minDCF is a more important evaluation metric than EER). Moreover, when dimension increases to 240, the advantages of multi-stream system becomes more obvious. So the proposed approach is more effective than simple increasing the dimension when the dimension is larger than the threshold, whereas less effective when the dimension is lower than the threshold. For this system, the threshold is around 120.

Feature Dim.	Stream Info[a]	minDCF	EER (%)	Optimal Weight[b]
3×32-d	Multi-stream	0.1805	2.489	[0.36, 0.31, 0.33]
96-d	FB-stream	0.1740	2.287	–
3×40-d	Multi-stream	0.1665	2.297	[0.39, 0.35, 0.26]
120-d	FB-stream	0.1720	2.245	–
3×80-d	Multi-stream	0.1367	1.946	[0.36, 0.32, 0.32]
240-d	FB-stream	0.1619	2.222	–

[a]FB-stream: $[20, 8\,000]$ Hz, LF-stream: $[20, 2\,000]$ Hz, HF-stream: $[1\,000, 8\,000]$ Hz. [b]From the left to the right, the weights are for FB-stream, LF-stream, and HF-stream, respectively.

Table 7. Evaluation results with the same feature dimension

The quantitative comparisons are illustrated in Figure 8 and Figure 9. Figure 8 shows the relative improvement of minDCF metric, where the left y-axis is evaluation result of minDCF with a different feature dimension, and the right y-axis is the percentage of a relative improvement of minDCF. The "original" means single-stream with the full frequency range, and it is displayed with "gray" color. The "multi-stream" means our proposed multi-stream with the full frequency selection, and it is displayed with "dark green" color. The red dash line illustrates the percentage of a relative improvement, and the value are 22.63 %, 20.53 %, and 23.20 % for 32-d, 40-d and 80-d, respectively.

Figure 9 shows the relative improvement of EER metric, where the left y-axis is evaluation result of EER with different feature dimension, and the right y-axis is the percentage of the relative improvement of EER. The "original" and "multi-stream" are displayed with "gray" and "olive" color, respectively. The red dash line illustrates the percentage of the relative improvement, and the value are 19.11 %, 15.77 %, and 15.28 % for 32-d, 40-d and 80-d, respectively.

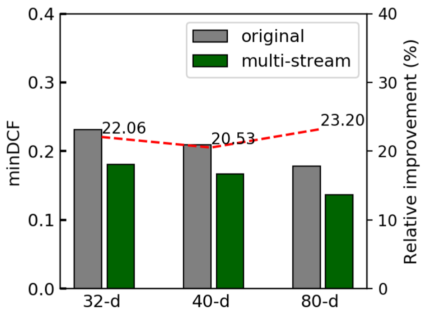


Figure 8. Relative improvement of minDCF

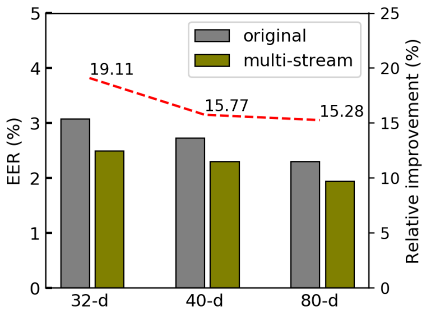


Figure 9. Relative improvement of EER

One remarkable thing is that the relative improvement of EER is not as large as minDCF because the objective of our proposed optimal weight search is based on minDCF. However, this algorithm can be applied to optimal weight search based on EER with few changes.

You might be thinking this is somewhat similar to the ensemble scenario, and wonder if it can outperform the fused system with different training methods, such as initialization method? To answer this question, we conduct additional experiments to make comparison between our proposed framework and the ensemble of three full-bandwidth models with different initialization methods.

We select three different initialization methods for comparison, which are the Kaiming method [56], Xavier method [57] and normal distribution. The experimental results of minDCF and EER under different training epochs are illustrated in Table 8 and Table 9, respectively. The term "SS" denotes as a Single Stream, whereas "MS" denotes as a Multiple Stream. For a fair comparison, we only change the initialization method of three full-bandwidth models, and all other training configurations remain the same. All five single stream systems are trained for 500 epochs and evaluated every 100 epochs.

System	100 Epochs	200 Epochs	300 Epochs	400 Epochs	500 Epochs
SS-1: FB-Kaiming-Init[a]	0.2028	0.1628	0.1688	0.1597	0.1606
SS-2: LF-Kaiming-Init	0.3620	0.3223	0.3256	0.3168	0.3125
SS-3: HF-Kaiming-Init	0.2574	0.2394	0.2237	0.2299	0.2195
SS-4: FB-Xavier-Init	0.2323	0.2144	0.1937	0.1825	0.1816
SS-5: FB-Normal-Init	0.3369	0.2933	0.2692	0.2611	0.2544
MS-1: SS-1, SS-4, SS-5	0.2193	0.1888	0.1819	0.1752	0.1722
Ours: SS-1, SS-2, SS-3	0.1609	0.1333	0.1381	0.1336	0.1310

[a]Baseline.

Table 8. Experimental results of minDCF under different training epochs

System	100 Epochs	200 Epochs	300 Epochs	400 Epochs	500 Epochs
SS-1: FB-Kaiming-Init[a]	2.792	2.350	2.244	2.239	2.132
SS-2: LF-Kaiming-Init	5.109	4.212	4.133	4.143	4.042
SS-3: HF-Kaiming-Init	3.691	3.085	3.021	2.872	2.802
SS-4: FB-Xavier-Init	2.840	2.553	2.239	2.096	2.106
SS-5: FB-Normal-Init	4.578	3.844	3.649	3.376	3.260
MS-1: SS-1, SS-4, SS-5	2.813	2.420	2.256	2.175	2.157
Ours: SS-1, SS-2, SS-3	2.303	1.919	1.866	1.834	1.781

[a]Baseline.

Table 9. Experimental results of EER with different training epochs

The system "SS-1: FB-Kaiming-Init" is defined as baseline, "SS-2" and "SS-3" are both initialized with Kaiming method and trained with partial bandwidth. "SS-4" and "SS-5" are full-bandwidth models with different initialization method. The system "MS-1" is the ensemble of "SS-1", "SS-4", and "SS-5", which is set to be the comparison object.

It can be seen from the table that our proposed method has a significant improvement over the whole training period, whereas the ensemble version has almost no improvement but slight degradation. The graphic comparison can be obtained in Figure 10 and Figure 11. Since "MS-1" has no improvement, we only plot the relative improvement of our proposed method as shown in Figure 12.

The experimental results show that the ensemble of three full-bandwidth models has poorer performance compared to single-stream baseline, whereas our proposed framework has at least 16.3 % and 16.5 % relative reduction in minDCF and EER, respectively.

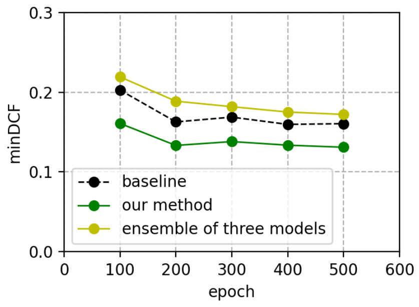


Figure 10. Comparison of minDCF

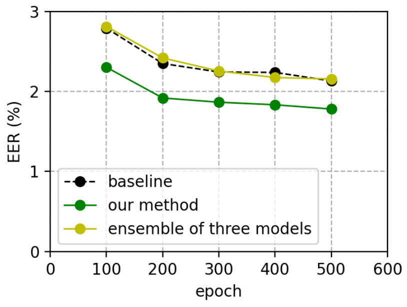


Figure 11. Comparison of EER

5 FUTURE WORK

It is obvious that applying frequency selection into speaker recognition with multi-stream can improve the performance. But the disadvantage is also obvious due to longer training time and larger network size.

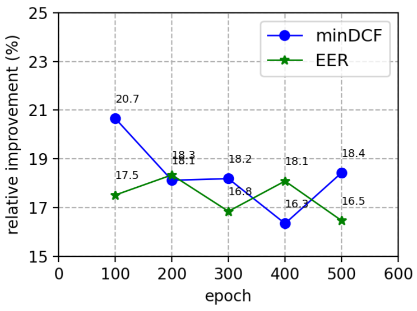


Figure 12. Relative improvement

As a follow-up, can we come up with a trade-off design between size and performance after investigating the probability of reducing size meanwhile yielding competitive benefits? More generally, is frequency selection also effective using other acoustic features such as MFCC, and other feature-extracting related techniques such as amplitude modulation – frequency modulation (AM-FM) technique [58], and other neural networks such as Recurrent Neural Network (RNN)? Furthermore, since the MFBE feature is obtained on human perception experiments, is it possible to find better features based on machine perception experiments? These are some remaining questions for us to explore and answer in the future.

6 CONCLUSION

In this paper, we propose a novel neural network framework based on frequency selection, namely multi-stream CNN, for robust speaker verification. The idea behind this proposal is that the diversity in temporal embeddings across multiple streams, where each stream process "partial" features extracted within a selected frequency range in parallel, could enhance the robustness of acoustic modeling and hence improve the overall performance. This approach can also be seen as an ensemble of three different models with different bandwidths. It is common knowledge that ensembles of many models usually increase the performance. Moreover, to address the problem of massive computation and memory requirements, we propose a more practical two-stage training method consisting of "Stage 1: Sequential training of each stream" and "Stage 2: Searching for optimal fusion weight".

To validate our proposed method, we conduct various training and testing experiments using the PyTorch library based on the VoxCeleb dataset and make a comprehensive comparison of the experimental results. The experimental results demonstrate the efficacy of our proposed method. The technique derived in our paper can be treated as a variant of metric learning for speaker verification.

Acknowledgement

This research was funded by the Zhejiang Provincial Natural Science Foundation of China under grant No. LGG21E060001.

REFERENCES

[1] Subramanian, A. S.—Wang, X.—Baskar, M. K.—Watanabe, S.—Taniguchi, T.—Tran, D.—Fujita, Y.: Speech Enhancement Using End-to-End Speech Recognition Objectives. 2019 IEEE Workshop on Applications of

Signal Processing to Audio and Acoustics (WASPAA), 2019, pp. 234–238, doi: 10.1109/WASPAA.2019.8937250.

[2] Zhang, W.—Chang, X.—Qian, Y.—Watanabe, S.: Improving End-to-End Single-Channel Multi-Talker Speech Recognition. IEEE/ACM Transactions on Audio, Speech, and Language Processing, Vol. 28, 2020, pp. 1385–1394, doi: 10.1109/TASLP.2020.2988423.

[3] Villalba, J.—Chen, N.—Snyder, D.—Garcia-Romero, D.—McCree, A. et al.: State-of-the-Art Speaker Recognition for Telephone and Video Speech: The JHU-MIT Submission for NIST SRE18. Proceedings of Interspeech 2019, 2019, pp. 1488–1492, doi: 10.21437/Interspeech.2019-2713.

[4] Xie, W.—Nagrani, A.—Chung, J. S.—Zisserman, A.: Utterance-Level Aggregation for Speaker Recognition in the Wild. ICASSP 2019, 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, 2019, pp. 5791–5795, doi: 10.1109/ICASSP.2019.8683120.

[5] Zhou, T.—Zhao, Y.—Li, J.—Gong, Y.—Wu, J.: CNN with Phonetic Attention for Text-Independent Speaker Verification. 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019, pp. 718–725, doi: 10.1109/ASRU46091.2019.9003826.

[6] Chung, J. S.—Huh, J.—Mun, S.—Lee, M.—Heo, H. S.—Choe, S.—Ham, C.—Jung, S.—Lee, B. J.—Han, I.: In Defence of Metric Learning for Speaker Recognition. Proceedings of Interspeech 2020, 2020, pp. 2977–2981, doi: 10.21437/Interspeech.2020-1064.

[7] Fujita, Y.—Kanda, N.—Horiguchi, S.—Xue, Y.—Nagamatsu, K.—Watanabe, S.: End-to-End Neural Speaker Diarization with Self-Attention. 2019 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU), 2019, pp. 296–303, doi: 10.1109/ASRU46091.2019.9003959.

[8] Huang, Z.—Watanabe, S.—Fujita, Y.—García, P.—Shao, Y.—Povey, D.—Khudanpur, S.: Speaker Diarization with Region Proposal Network. ICASSP 2020, 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, 2020, pp. 6514–6518, doi: 10.1109/ICASSP40776.2020.9053760.

[9] Snyder, D.—Garcia-Romero, D.—Sell, G.—Povey, D.—Khudanpur, S.: X-Vectors: Robust DNN Embeddings for Speaker Recognition. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 5329–5333, doi: 10.1109/ICASSP.2018.8461375.

[10] Moritz, N.—Hori, T.—Roux, J. L.: Semi-Supervised Speech Recognition via Graph-Based Temporal Classification. ICASSP 2021, 2021 IEEE International Conference on Acoustics, Speech and Signal Processing, 2021, pp. 6548–6552, doi: 10.1109/ICASSP39728.2021.9414058.

[11] Nagrani, A.—Chung, J. S.—Albanie, S.—Zisserman, A.: Disentangled Speech Embeddings Using Cross-Modal Self-Supervision. ICASSP 2020, 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, 2020, pp. 6829–6833, doi: 10.1109/ICASSP40776.2020.9054057.

[12] LeCun, Y.—Bengio, Y.—Hinton, G.: Deep Learning. Nature, Vol. 521, 2015, No. 7553, pp. 436–444, doi: 10.1038/nature14539.

[13] Chung, J. S.—Nagrani, A.—Coto, E.—Xie, W.—McLaren, M.—Reynolds, D. A.—Zisserman, A.: VoxSRC 2019: The First VoxCeleb Speaker Recognition Challenge. CoRR, 2019, doi: 10.48550/arXiv.1912.02522.

[14] Peddinti, V.—Povey, D.—Khudanpur, S.: A Time Delay Neural Network Architecture for Efficient Modeling of Long Temporal Contexts. Proceedings of Interspeech 2015, 2015, pp. 3214–3218, doi: 10.21437/Interspeech.2015-647.

[15] Li, W.—Wang, L.—Zhou, Y.—Dines, J.—Magimai-Doss, M.—Bourlard, H.—Liao, Q.: Feature Mapping of Multiple Beamformed Sources for Robust Overlapping Speech Recognition Using a Microphone Array. IEEE/ACM Transactions on Audio, Speech, and Language Processing, Vol. 22, 2014, No. 12, pp. 2244–2255, doi: 10.1109/TASLP.2014.2364130.

[16] Huang, C. W.—Narayanan, S. S.: Deep Convolutional Recurrent Neural Network with Attention Mechanism for Robust Speech Emotion Recognition. 2017 IEEE International Conference on Multimedia and Expo (ICME), 2017, pp. 583–588, doi: 10.1109/ICME.2017.8019296.

[17] Likitha, M. S.—Gupta, S. R. R.—Hasitha, K.—Raju, A. U.: Speech Based Human Emotion Recognition Using MFCC. 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2017, pp. 2257–2260, doi: 10.1109/WiSPNET.2017.8300161.

[18] Juvela, L.—Bollepalli, B.—Wang, X.—Kameoka, H.—Airaksinen, M.—Yamagishi, J.—Alku, P.: Speech Waveform Synthesis from MFCC Sequences with Generative Adversarial Networks. 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2018, pp. 5679–5683, doi: 10.1109/ICASSP.2018.8461852.

[19] Li, X.—Zhong, J.—Wu, X.—Yu, J.—Liu, X.—Meng, H.: Adversarial Attacks on GMM I-Vector Based Speaker Verification Systems. ICASSP 2020, 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, 2020, pp. 6579–6583, doi: 10.1109/ICASSP40776.2020.9053076.

[20] Zeinali, H.—Wang, S.—Silnova, A.—Matějka, P.—Plchot, O.: BUT System Description to VoxCeleb Speaker Recognition Challenge 2019. CoRR, 2019, doi: 10.48550/arXiv.1910.12592.

[21] Ramoji, S.—Mohan, A.—Mysore, B.—Bhatia, A.—Singh, P.—Vardhan, H.—Ganapathy, S.: The LEAP Speaker Recognition System for NIST SRE 2018 Challenge. ICASSP 2019, 2019 IEEE International Conference on Acoustics, Speech and Signal Processing, 2019, pp. 5771–5775, doi: 10.1109/ICASSP.2019.8683148.

[22] Ju, F.—Sun, Y.—Gao, J.—Hu, Y.—Yin, B.: Probabilistic Linear Discriminant Analysis with Vectorial Representation for Tensor Data. IEEE Transactions on Neural Networks and Learning Systems, Vol. 30, 2019, No. 10, pp. 2938–2950, doi: 10.1109/TNNLS.2019.2901309.

[23] Ramoji, S.—Krishnan, P.—Mysore, B.—Singh, P.—Ganapathy, S.: LEAP System for SRE19 CTS Challenge – Improvements and Error Analysis. Proceedings of the Speaker and Language Recognition Workshop (Odyssey 2020), 2020, pp. 281–288, doi: 10.21437/Odyssey.2020-40.

[24] Tu, Z.—Xie, W.—Qin, Q.—Poppe, R.—Veltkamp, R. C.—Li, B.—Yuan, J.: Multi-Stream CNN: Learning Representations Based on Human-Related Regions for Action Recognition. Pattern Recognition, Vol. 79, 2018, pp. 32–43, doi: 10.1016/j.patcog.2018.01.020.

[25] Wei, W.—Wong, Y.—Du, Y.—Hu, Y.—Kankanhalli, M.—Geng, W.: A Multi-Stream Convolutional Neural Network for sEMG-Based Gesture Recognition in Muscle-Computer Interface. Pattern Recognition Letters, Vol. 119, 2019, pp. 131–138, doi: 10.1016/j.patrec.2017.12.005.

[26] Kuang, Q.—Jin, X.—Zhao, Q.—Zhou, B.: Deep Multimodality Learning for UAV Video Aesthetic Quality Assessment. IEEE Transactions on Multimedia, Vol. 22, 2020, No. 10, pp. 2623–2634, doi: 10.1109/TMM.2019.2960656.

[27] Ali, A.—Renals, S.: Word Error Rate Estimation Without ASR Output: e-WER2. Proceedings of Interspeech 2020, 2020, pp. 616–620, doi: 10.21437/Interspeech.2020-2357.

[28] Yu, W.—Zeiler, S.—Kolossa, D.: Multimodal Integration for Large-Vocabulary Audio-Visual Speech Recognition. 2020 28th European Signal Processing Conference (EUSIPCO), 2021, pp. 341–345, doi: 10.23919/Eusipco47968.2020.9287841.

[29] Wei, L.—Zhang, J.—Hou, J.—Dai, L.: Attentive Fusion Enhanced Audio-Visual Encoding for Transformer Based Robust Speech Recognition. 2020 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), 2020, pp. 638–643, doi: 10.48550/arXiv.2008.02686.

[30] Ghorbani, S.—Gaur, Y.—Shi, Y.—Li, J.: Listen, Look and Deliberate: Visual Context-Aware Speech Recognition Using Pre-Trained Text-Video Representations. 2021 IEEE Spoken Language Technology Workshop (SLT), 2021, pp. 621–628, doi: 10.1109/SLT48900.2021.9383466.

[31] Shon, S.—Glass, J.: Multimodal Association for Speaker Verification. Proceedings of Interspeech 2020, 2020, pp. 2247–2251, doi: 10.21437/Interspeech.2020-1996.

[32] Li, R.—Wang, X.—Mallidi, S. H.—Watanabe, S.—Hori, T.—Hermansky, H.: Multi-Stream End-to-End Speech Recognition. IEEE/ACM Transactions on Audio, Speech, and Language Processing, Vol. 28, 2020, pp. 646–655, doi: 10.1109/TASLP.2019.2959721.

[33] Li, R.—Sell, G.—Wang, X.—Watanabe, S.—Hermansky, H.: A Practical Two-Stage Training Strategy for Multi-Stream End-to-End Speech Recognition. ICASSP 2020, 2020 IEEE International Conference on Acoustics, Speech and Signal Processing, 2020, pp. 7014–7018, doi: 10.1109/ICASSP40776.2020.9053455.

[34] Nemala, S. K.—Elhilali, M.: Multistream Robust Speaker Recognition Based on Speech Intelligibility. 2011 45th Annual Conference on Information Sciences and Systems, 2011, pp. 1–5, doi: 10.1109/CISS.2011.5766105.

[35] Nemala, S. K.—Patil, K.—Elhilali, M.: A Multistream Feature Framework Based on Bandpass Modulation Filtering for Robust Speech Recognition. IEEE Transactions on Audio, Speech, and Language Processing, Vol. 21, 2013, No. 2, pp. 416–426, doi: 10.1109/TASL.2012.2219526.

[36] Mallidi, S. H.—Hermansky, H.: Novel Neural Network Based Fusion for Multistream ASR. 2016 IEEE International Conference on Acous-

tics, Speech and Signal Processing (ICASSP), 2016, pp. 5680–5684, doi: 10.1109/ICASSP.2016.7472765.

[37] Pan, J.—Shapiro, J.—Wohlwend, J.—Han, K. J.—Lei, T.—Ma, T.: ASAPP-ASR: Multistream CNN and Self-Attentive SRU for SOTA Speech Recognition. Proceedings of Interspeech 2020, 2020, pp. 16–20, doi: 10.21437/Interspeech.2020-2947.

[38] Facts About Speech Intelligibility. DPA Microphones, 2021, `https://www.dpamicrophones.com/mic-university/facts-about-speech-intelligibility` (Accessed: 2021-03-03).

[39] Paszke, A.—Gross, S.—Massa, F.—Lerer, A.—Bradbury, J. et al.: PyTorch: An Imperative Style, High-Performance Deep Learning Library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (Eds.): Advances in Neural Information Processing Systems 32 (NeurIPS 2019). Curran Associates, Inc., 2019, pp. 8026–8037, `https://proceedings.neurips.cc/paper_files/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf`.

[40] Fayek, H. M.: Speech Processing for Machine Learning: Filter Banks, Mel-Frequency Cepstral Coefficients (MFCCs) and What's In-Between. 2016, `http://www.haythamfayek.com/2016/04/21/speech-processing-for-machine-learning.html` (Accessed: 2024-08-02).

[41] Tak, R. N.—Agrawal, D. M.—Patil, H. A.: Novel Phase Encoded Mel Filterbank Energies for Environmental Sound Classification. In: Shankar, B. U., Ghosh, K., Mandal, D. P., Ray, S. S., Zhang, D., Pal, S. K. (Eds.): Pattern Recognition and Machine Intelligence (PReMI 2017). Springer, Cham, Lecture Notes in Computer Science, Vol. 10597, 2015, pp. 317–325, doi: 10.1007/978-3-319-69900-4_40.

[42] Meng, H.—Yan, T.—Yuan, F.—Wei, H.: Speech Emotion Recognition from 3D Log-Mel Spectrograms with Deep Learning Network. IEEE Access, Vol. 7, 2019, pp. 125865–125881, doi: 10.1109/ACCESS.2019.2938007.

[43] TorchAudio Tutorials. 2019, `https://pytorch.org/audio/stable/functional.html` (Accessed: 2024-08-02).

[44] He, K.—Zhang, X.—Ren, S.—Sun, J.: Deep Residual Learning for Image Recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778, doi: 10.1109/CVPR.2016.90.

[45] Bai, Z.—Zhang, X. L.—Chen, J.: Cosine Metric Learning for Speaker Verification in the i-Vector Space. Proceedings of Interspeech 2018, 2018, pp. 1126–1130, doi: 10.21437/Interspeech.2018-1593.

[46] Nagrani, A.—Chung, J. S.—Xie, W.—Zisserman, A.: VoxCeleb: Large-Scale Speaker Verification in the Wild. Computer Speech & Language, Vol. 60, 2020, Art. No. 101027, doi: 10.1016/j.csl.2019.101027.

[47] Nagrani, A.—Chung, J. S.—Zisserman, A.: VoxCeleb: A Large-Scale Speaker Identification Dataset. Proceedings of Interspeech 2017, 2017, pp. 2616–2620, doi: 10.21437/Interspeech.2017-950.

[48] Chung, J. S.—Nagrani, A.—Zisserman, A.: VoxCeleb2: Deep Speaker Recognition. Proceedings of Interspeech 2018, 2018, pp. 1086–1090, doi: 10.21437/Interspeech.2018-1929.

[49] Chen, S.: ShaneRun System Description to VoxCeleb Speaker Recognition Challenge 2020. CoRR, 2020, doi: 10.48550/arXiv.2011.01518.

[50] van der Maaten, L.—Hinton, G.: Visualizing Data Using t-SNE. Journal of Machine Learning Research, Vol. 9, 2008, No. 86, pp. 2579–2605, `http://jmlr.org/papers/v9/vandermaaten08a.html`.

[51] 2018 NIST Speaker Recognition Evaluation. NIST, 2019, `https://www.nist.gov/system/files/documents/2018/08/17/sre18_eval_plan_2018-05-31_v6.pdf` (Accessed: 2024-08-02).

[52] Matějka, P.—Plchot, O.—Glembek, O.—Burget, L.—Rohdin, J.—Zeinali, H.—Mošner, L.—Silnova, A.—Novotný, O.—Diez, M.—Černocký, J.: 13 Years of Speaker Recognition Research at BUT with Longitudinal Analysis of NIST SRE. Computer Speech & Language, Vol. 63, 2020, Art. No. 101035, doi: 10.1016/j.csl.2019.101035.

[53] Dehak, N.—Kenny, P. J.—Dehak, R.—Dumouchel, P.—Ouellet, P.: Front-End Factor Analysis for Speaker Verification. IEEE Transactions on Audio, Speech, and Language Processing, Vol. 20, 2011, No. 4, pp. 788–798, doi: 10.1109/TASL.2010.2064307.

[54] Jung, Y.—Kim, Y.—Lim, H.—Choi, Y.—Kim, H.: Spatial Pyramid Encoding with Graph-Based Normalization for Text-Independent Speaker Verification. Proceedings of Interspeech 2019, 2019, pp. 4030–4034, doi: 10.21437/Interspeech.2019-2177.

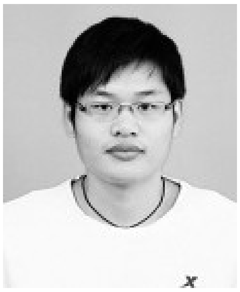
[55] Kye, S. M.—Jung, Y.—Lee, H. B.—Hwang, S. J.—Kim, H.: Meta-Learning for Short Utterance Speaker Recognition with Imbalance Length Pairs. Proceedings of Interspeech 2020, 2020, pp. 2982–2986, doi: 10.21437/Interspeech.2020-1283.

[56] He, K.—Zhang, X.—Ren, S.—Sun, J.: Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 1026–1034, doi: 10.1109/ICCV.2015.123.

[57] Glorot, X.—Bengio, Y.: Understanding the Difficulty of Training Deep Feedforward Neural Networks. In: Teh, Y. W., Titterington, M. (Eds.): Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. PMLR, Vol. 9, 2010, pp. 249–256, `https://proceedings.mlr.press/v9/glorot10a.html`.

[58] Deshpande, M. S.—Holambe, R. S.: Robust Speaker Identification in Babble Noise. Proceedings of the International Conference & Workshop on Emerging Trends in Technology, 2011, pp. 635–640, doi: 10.1145/1980022.1980160.

Wei Yao works as Associate Professor with the College of Electric Engineering, Zhejiang University of Water Resources and Electric Power. He received his M.Sc. degree in control theory and control engineering from the Zhejiang University of Technology and his Ph.D. degree in electrical engineering from the Zhejiang University, in 2002 and 2015 respectively. His research interests include speech recognition, image processing, embedded systems, power electronics, etc.



Shen Chen works for Wanbang Digital Energy Co., Ltd., China, as Product Director. He is currently the Product Director of the residential energy storage team and holds more than thirty patents. His research interests include the Internet of Things, energy management system, speech enhancement, speaker recognition, etc.

Jiamin Cui works as a teacher at the College of Electrical Engineering, Zhejiang University of Water Resources and Electric Power. His research interests include communication power supply, intelligent battery charger, machine learning in power electronics, etc.

Yaolin Lou works for the College of Electrical Engineering, Zhejiang University of Water Resources and Electric Power since 2019. His research interests are new energy science and engineering, wind power controller system design and simulation, and the electric power system with renewable energy.