SOFTWARE FOR RECOGNIZING SPEAKER BY VOICE

Chen Zhengyu

Group 467311

Belarusian State University of Informatics and Radioelectronics[1], Minsk, Republic of Belarus

Zelmansky O.B. – Associate Professor of the Department of Information Security, PhD, Associate Professor.

Annotation. FBank (Filter Bank) is a front-end processing algorithm that processes audio in a way similar to the human ear and extracts features to improve the performance of speech recognition. The system uses an efficient context-aware masking-based network, CAM++, which uses a densely connected time-delay neural network (D-TDNN) as the backbone and adopts a novel multi-granularity pooling to capture different levels of context information.Based on the respective advantages of FBank and CAM++ models, this study designs a software for recognizing speaker by voice and implements the system through pytorch.

Keywords. speaker recognition, feature Extraction, neural networks, pytorch

Introduction. Speaker recognition (SR) is a biometric technology that identifies individuals based on unique vocal characteristics. In recent years, with the rapid development of deep learning technology, speaker recognition technology has become more and more widely used, for example call center operation, criminal investigation, smart speaker and robotics, security for phone and other fields.

This thesis focuses on the design and implementation of software for recognizing speaker by voice. This involves two main aspects: one is selecting and training the model, and the other is designing the front-end interface.This system is based on the CAM++ model.[1] It includes a convolutional module as the front end and a D-TDNN as the backbone. Each D-TDNN layer has an improved context-aware mechanism built in, including multi-granularity pooling to capture speaker characteristics. This reduces the amount of computation and increases the inference speed. PyQt5 is then used to design an interface that implements registration, recording, recognition, and other functions, connecting it to the model.

Before training the model, FBank is used to extract features from the training data to simulate the characteristics of the human ear to improve the accuracy of the model. Experiments were conducted on the CN-Celeb database of about 1,000 speakers. The preprocessing method used was FBank, and the loss function was Angular additive margin softmax (AAM-Softmax) loss. The EER of the CAM++ model was 14.73% and the MinDCF was 0.41999.

Main Part. In the SR (Speaker Recognition) system, the first step is to extract features. The most commonly used features in speech-related tasks are MFCC and FBank. This system chooses to use FBank to extract features from data. The general steps to obtain the FBank features of speech signals are: pre-emphasis, framing, windowing, short-time Fourier transform (STFT), Mel filtering, etc.

Pre-emphasis strengthens high frequencies because the human vocal tract radiates sound waves through air, which acts as a signal carrier that both transmits and dissipates energy. As the medium is the carrier of sound energy, when the size of the sound source is constant, the higher the frequency, the more serious the loss of sound energy by the medium. It is necessary to divide the audio of indefinite length into small segments of fixed length. This step is called framing. Generally, 10-30ms is taken as one frame. In order to avoid missing

the signal at the window boundary, there should be frame overlap when offsetting the frame (frames need to overlap a part). This can avoid too much change in characteristics between frames. The usual choice is 25ms per frame, and the frame shift is 10ms. For example, our system uses 80-dimensional FBank features extracted over a 25 ms long window for every 10 ms as input. After framing, the signal is still in the time domain. In order to extract FBank features, we first need to convert the time domain signal into the frequency domain signal. Fourier transform can convert the signal from the time domain to the frequency domain. Fourier transform can be divided into continuous Fourier transform and discrete Fourier transform. Because this system uses digital audio (not analog audio), discrete Fourier transform is used. The discrete Fourier transform is calculated as follows:

$$S_i(k) = \sum_{n=1}^{N} x(n) e^{-j2\pi kn/N} \qquad 1 \le k \le K$$

Where $x(n)$ – audio sampling point, N – Number of sampling points.

The specific relationship between Mel frequency and actual frequency is as follows:

$$f_{mel}(f) = 2595 \cdot log(1 + \frac{f}{700Hz})$$

The hearing characteristics of the human ear are consistent with the growth of the Mel frequency. The actual frequency is linear below 1000Hz and logarithmic above 1000Hz.

Usually we set the upper and lower limits of the frequency to shield some unnecessary or noisy frequency ranges (the lower limit is usually set to around 20 Hz, and the upper limit is half of the audio sampling rate), and convert it to Mel frequency. Then configure a triangular filter bank of K channels on the Mel frequency axis, and K is generally 40. The triangular window function is:

$$H_m(k) = \begin{cases} \dfrac{k - f_{m-1}}{f_m - f_{m-1}} & f_{m-1} \le k < f_m \\ \dfrac{f_{m+1} - k}{f_{m+1} - f_m} & f_m \le k < f_{m+1} \\ 0 & otherwise \end{cases}$$

Where f_m – Center frequency of the m filter, k – Frequency index after FFT.

The system's input consisted of 80-bin FBank features computed with a 25 ms window size and 10 ms frame shift. In addition, two types of data augmentation are applied simultaneously during model training: synthesizing reverberation effects through room impulse response (RIR)[2], and adding environmental noise using the MUSAN corpus [3]. Speech, music, and noise from the MUSAN database and RIR are randomly added to the original training data. The training dataset is augmented twofold by synthesizing reverberation effects (RIR) and adding environmental noise (from MUSAN). This increases the amount and diversity of the existing training data, and achieves a significant improvement for the system[4].

The proposed CAM++ architecture adopts a dual-module design consisting of a front-end convolution module (FCM) and a D-TDNN backbone. The FCM employs multiple 2D convolutional blocks with residual connections to effectively capture high-resolution time-frequency representations from the acoustic features. These feature maps are then flattened along both channel and frequency dimensions before being fed into the subsequent D-TDNN backbone. The backbone comprises three sequential blocks, each containing stacked D-TDNN layers enhanced with improved CAM modules. These modules dynamically assign attention weights to

the TDNN layer outputs, enabling network-wide context-aware feature refinement. This hierarchical architecture allows for comprehensive processing of spectro-temporal patterns while maintaining efficient feature abstraction.

AAM-Softmax (Additive Angular Margin Softmax) is a loss function widely used in speaker recognition and facial recognition[5]. It is an improved method of Large Margin Softmax series. Its core idea is to increase the difference between classes and improve the discriminability of features by introducing margin in the angular space.The formula of AAM-Softmax is as follows:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N} log\frac{e^{s\cdot(cos(\theta_{y_i}+m))}}{e^{s\cdot(cos(\theta_{y_i}+m))}+\sum_{j=1,j\neq y_i}^{n} e^{e\cdot cos\theta_j}}$$

Where θ_{y_i} – the angle between the sample x_i and the weight vector W_{y_i} of its true category y_i. s – a scaling factor used to control the effect of the modulus of the feature vector on the loss. m – the angle margin, which is used to increase the classification difficulty of samples near the decision boundary.

The model evaluation metrics are Equal Error Rate (EER) and Minimum Detection Cost (MinDCF). EER does not rely on manually set thresholds and directly reflects the inherent distinguishing ability of the model. EER is the error rate when the False Acceptance Rate (FAR) and the False Rejection Rate (FRR) are equal, that is:

$$EER = FAR = FRR$$

MinDCF is the value that minimizes the detection cost function (DCF).MinDCF is the minimum value of DCF under all possible thresholds θ. By adjusting the $C_{Miss}/C_{FalseAlarm}$ ratio, it can adapt to different scene requirements. The formula is as follows:

$$C_{Det}(\theta) = C_{Miss} \times P_{Target} \times P_{Miss}(\theta) + C_{FalseAlarm} \times (1 - P_{Target}) \times P_{FalseAlarm}(\theta)$$

Where θ – decision threshold, C_{Miss} – Cost of missed judgment, $C_{FalseAlarm}$ – The cost of misjudgment, P_{Target} – a priori probability of the specified target speaker.

The prediction module can receive audio, resample the audio (16kHz), normalize, extract features. Using a pre-trained neural network model, a fixed-dimensional speaker recognition vector is extracted. That is, the feature data is put into the model for inference to obtain a 256-dimensional vector representing the speaker's voiceprint features, which is stored in the audio database together with the label. This system uses cosine similarity to calculate the score. The formula is as follows：

$$similarity = cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}}$$

If the similarity exceeds the set threshold, they are determined to be the same speaker. In speaker recognition, the system will select the speaker with the highest score and exceeding the threshold as the recognition result. If it does not exceed the threshold, it will be judged as not recognizing the registered user. In addition, this system is designed so that each user can register multiple voice samples, and the system will calculate the average features to improve recognition robustness.

The front-end interface is designed by pyqt5. Users can register speaker recognition information or perform speaker recognition by recording through the interface. At the same time, the recording length and judgment threshold can be set. The recognized registration name and confidence will be returned to the

interface. Audio files can also be recognized. By passing the path of the audio file into the system, the system can recognize it and return the result.

Conclusion. This thesis introduces the design and implementation of a software for recognizing speaker by voice, which combines deep learning techniques with a PyQt5 front-end interface to achieve robust speaker recognition through FBank feature extraction, data augmentation, CAM++ model and an optimized AAM-Softmax loss function. The system shows competitive performance on the CN-Celeb dataset with an EER of 14.73% and a MinDCF of 0.41999, using a combination of convolutional front-end modules and context-aware D-TDNN layers of the CAM++ architecture to effectively capture speaker features while reducing computational overhead. By using FBank features to simulate human auditory perception and enhancing model generalization through data augmentation techniques such as RIR reverberation and MUSAN noise, the system supports practical applications such as speech registration, real-time recognition, and audio file analysis, and adopts a threshold-based speaker verification decision mechanism. The PyQt5 interface provides seamless interaction with configurable parameters for recording, registration, and recognition, thereby fulfilling all software functional requirements.

List of sources used:

1. CAM++: A fast and efficient network for speaker verification using context-aware masking / H. Wang, S. Zheng, Y. Chen [et al.] // arXiv preprint arXiv:2303.00332, 2023..

2. Musan: A music, speech, and noise corpus / D. Snyder, G. Chen, D. Povey [et al.] // arXiv preprint arXiv:1510.08484, 2015.

3. A study on data augmentation of reverberant speech for robust speech recognition / T. Ko, V. Peddinti, D. Povey [et al.] // 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). - P. 5220-5224.

4. X-vectors: Robust DNN embeddings for speaker recognition / D. Snyder, D. Garcia-Romero, G. Sell [et al.] // 2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). - P. 5329-5333.

5. Deng J, Guo J, Xue N, et al. Arcface: Additive angular margin loss for deep face recognition[C]//Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2019: 4690-4699.