ISSN: 2688-8645

Speaker Recognition System based on Triplet State Loss Function

Tingyu Li

College of Computer Science and Engineering, Northeastern University, Shenyang, 110819, China

20203465@stu.neu.edu.cn

Abstract

The purpose of this paper is to build a model and design a speaker recognition system by comprehensively summarizing and learning the research data of speaker speech recognition models at home and abroad, and adopting a research method based on deep machine learning theory. Its main contents and proposed methods are as follows: For data processing, firstly, select and download the public data set from official website, preprocess each voice in the data set, extract Fbank features, convert it into. npy, store it in a file, process the voice into a format suitable for model input, and wait for subsequent input into the model. In practice, a ResCNN architecture based on convolution neural network is used to build a model. The model uses triplet loss function training to map speech to hyperplane, so cosine similarity is directly used to characterize the distance between two speakers. Speaker verification function provides three different ways to obtain speech, input the two acquired speech into the model, judge the similarity of the two speech and give the judgment result. For the speaker recognition model, three different ways can also be used to obtain the speech and determine which speaker the speech is in the corpus. For the speaker confirmation model, a speech is randomly played, and a speaker is randomly selected to judge whether the speech is the speaker's voice.

Keywords

ResCNN; Triplet Loss Function; Fbank Features; Cosine Similarity.

1. Introduction

Speaker recognition algorithms try to determine the identity of speakers from audio. Two common recognition tasks are speaker authentication (to determine whether the speaker's claimed identity is true or false) and speaker identification (to classify the identities of unknown voices in a group of speakers). In all these tasks, embedding methods can be used to map utterances to feature spaces whose distances correspond to speaker similarity. The whole process can be divided into: data preprocessing, model training, generation of speaker audio feature library and function realization.

1.1. Data Pre-processing

Download the train-clean-100 data set from openslr official website, extract the path of the data set and the corresponding speaker tag, and organize them into DataFrame data structure. Because of the large amount of data, the calling process pool divides the data into five processes, and processes the data separately in different processes. The data processing method for each speech is as follows: firstly, the wav file is loaded, the speech is read, the mute part is deleted by VAD method, the 64-dimensional Fbank feature of the non-mute data part is extracted, and then the data is normalized to make its mean value zero and variance one, and finally stored as the corresponding npy format file.

ISSN: 2688-8645

The speech endpoint detection method uses short-term energy method to judge the relationship between speech energy value and threshold value in a period of time. If it is greater than threshold value, it is speech segment, otherwise, it is mute segment

The speaker id is extracted from the npy file and stored in the program by increasing in positive order, such as the id is "61-70970-0006", the label in the program is 61, the id is "3729-6852-0004", and the label in the program is 3729. The format entered into the model is a single hot-coded format, which is marked with 1 in this speaker position and 0 in other positions.

1.2. Model Training

The residual neural network is used to extract the audio features of each speech and generate the corresponding embedding vectors. The model is trained by maximizing the cosine similarity of audio embedding pairs from the same speaker and minimizing the cosine similarity of audio embedding pairs from different speakers through triplet loss function. Before formal training, softmax and cross entropy loss are introduced to pre-train the model. The network weights initialized with softmax pre-training have the advantage of more stable convergence.

1.3. Audio Feature Library

The trained model is used to reason all the audio of each speaker in the data set, and each audio corresponds to an embedded vector. All the embedded vectors of each speaker are averaged, and the final result is saved as the vector feature of each speaker.

1.4. Function Implementation

We have realized three functions: speaker identification, confirmation and verification. Speaker recognition refers to the process of identifying which speaker in the data set is the voice to be recognized, that is, the process of finding the target speaker in the data set. Speaker confirmation refers to specifying a speaker and judging whether the speaker's voice is the claimed voice. Speaker verification refers to inputting two sounds and judging whether these two sounds are the sounds made by the same speaker.

2. Related Works

2.1. Approaches

2.1.1. Structure

Table 1. Network structure

layer name	structure	stride	dim	# params
conv64-s	$5 \times 5, 64$	2×2	2048	6K
res64	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	1×1	2048	41K×6
conv128-s	$5 \times 5, 128$	2×2	2048	209K
res128	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 3$	1×1	2048	151K×6
conv256-s	$5 \times 5, 256$	2×2	2048	823K
res256	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 3$	1×1	2048	594K×6
conv512-s	$5 \times 5, 512$	2×2	2048	3.3M
res512	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	1×1	2048	2.4M×6
average	-	-	2048	0
affine	2048×512	-	512	1M
ln	-	-	512	0
triplet	-	-	512	0
total				24M

The structure of the whole model is as follows: in the first batch of network layer, fbank features extracted from speech are input into convolution layer with 64 channels, 5*5 size and 2*2 step

ISSN: 2688-8645

size, then the processing results are input into ReLU layer, and finally in the batch of network layer, three residual blocks with 64 channels, 3*3 size and 1*1 step size are input in turn. The processing data is then entered into the network layer of the second batch, where the difference is that the number of channels is 128, and the rest is the same as the network structure of the previous batch. In the third batch network layer, the number of channels is 256, and in the fourth batch network layer, the number of channels is 512. The structure of the four batches of networks is the same, but the difference lies in the different number of channels. (Batch Normalization).

Is adopted between each convolution layer and nonlinear layer. After four batches of network layer processing, the data is input to the average layer, and a piece of speech corresponds to a piece of features, thus achieving the purpose of transforming frame-level features into discourse-level features. In order to reduce the dimension of features, discourse-level features are input into affine layer, and the data dimension is reduced from 2048 to 512. After that, the data is input into the length normalization layer, and the temporary pool features are mapped to the speaker embedding.

2.1.2. Triplet Loss

The cosine similarity of two speech feature vector pairs is used to judge whether they are the same speaker, and the calculation formula is formula.

$$\cos(x_i, x_j) = x_i^T x_j$$

Where, x_i and x_j represent the feature vectors of two sounds respectively.

The loss function used in the training of the model is triple loss function, and three kinds of samples are taken as inputs, which are called positive samples, target samples and negative samples. Among them, the first two samples are different sounds from the same speaker, and the latter two samples are sounds from different speakers. The goal of training is to make the cosine similarity of the former two samples greater than that of the latter two samples, even if the samples of the same speaker are closer in space. As shown in the figure:

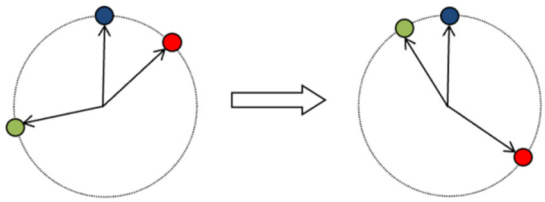

Fig 1. triplet loss function

That is, in the whole training process, I hope the model can satisfy the formula

$$s_i^{ap} - \alpha > s_i^{an}$$

Among α them, the discrimination threshold between positive samples and negative samples, and the cosine similarity between s_i^{ap} anchor samples and positive samples at the i^{th} calculation. It is the cosine similarity of s_i^{an} anchor sample and negative sample at the i^{th} calculation.

2.1.3. SoftMax Pre-training

In the pre-training stage, the structure of the model is different from that of the formal training model. At the end of the model network, a classification layer is used to replace the length normalization layer and triple loss layer at the end of ResCNN, and the cross-entropy loss function is used to update the model parameters. The purpose is to make the model converge faster.

ISSN: 2688-8645

2.2. Our Work

Preprocessing: A module that reads speech, filters mute, extracts fbank features, and saves them in. npy format

Pretraining: Pre-training of softmax classification is carried out, and the back end of the original model is changed into softmax classification layer

random_batch: Select a batch at random. Randomly select one from the data set as anchor, then randomly select one from the voice of the same speaker as positive, and then randomly select one from the voice of different speakers as negative. It is better to use random_batch at the early stage of training.

select_batch: Select the best batch to feed to the network.

function implementation: Using the trained model to create the audio feature vector of each speaker to realize various functional applications.

3. Dataset

From the librispeech data set, there are 251 speakers, with a total of 28,539 sounds, with a total duration of about 100 hours. Each person's voice is placed in a folder to distinguish different speakers, which is divided into voice training set and voice test set. The text materials of the recordings are mainly public audiobooks based on LibriVox.

4 Experiments

3.1. Select Batch

Choose the best batch to feed to the network. This is one of the cores of this experiment. Take some specific numbers as examples, and explain them as follows: First, choose 160seaker. Then read 2 speech features (fbank) from each speaker and put them into the candidate set. Then, the 320-candidate set is propagated forward through the network to obtain embeddings. Then put the 320 embedding in the history embedding table, and we access the 10-embedding data of history, so we have 3200 embedding of speech. Then select 4 speakers from 3200 voices, and calculate the similarity of embeddings between the voices of 4 speakers and other voices. Then, for each speaker, select the two most dissimilar positions and the two most similar negatives to form two pairs of pairs. So, you get eight pairs of anchor-positive-nagative pairs, and then you send this to the network for training.

3.2. Hyperparameter

Table 2 shows the Hyperparameters we set during training.

Table 2. Parameter

parameter	value
optimizer	adam
learning_rate	0.001
batch_size	32
candidates_pre_batch	320
save_per_epoch	200
alpha	0.2
sample_rate	16000
truncate_sound_seconds	(0.2,1.81)

ISSN: 2688-8645

3.3. Performance Analysis

When training the model, the computed value of judging the model performance index, such as equal error rate, accuracy rate, harmonic average value and so on, is saved in the file. After training, we use the pyplot function in matplotlib library to visualize the data and analyze the model.

3.3.1. Pretraining

In the pre-training stage, every time the parameters of the model are updated, the loss value and accuracy value of the model will be calculated by using the model. train_on_batch function, and the values will be analyzed after the training. Analyze the loss value and accuracy value, as shown in the figure:

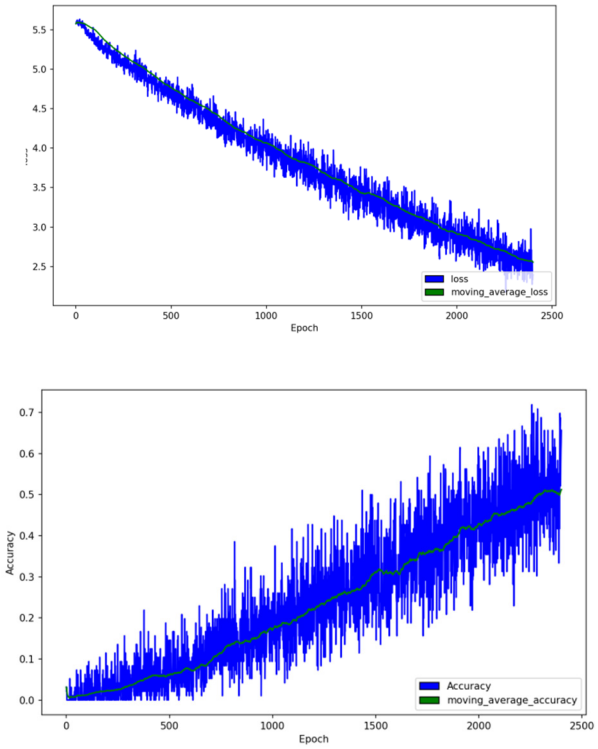

Fig 2. Loss value and accuracy value

The loss value tends to decline, while the accuracy value tends to rise.

3.3.2. Training

In the formal training stage, information such as f-measure, error and acquisition will be recorded, as shown in the figure:

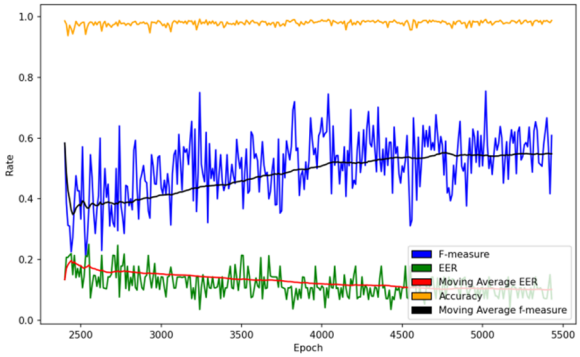

Fig 3. F-measure, error and acquisition

ISSN: 2688-8645

We find that loss rises at first and then falls, which may be caused by the transition from softmax-loss to triplet-loss. After that, f-measure basically increased steadily, while error basically decreased steadily. Except the average error value curve and the average f-measure curve are smooth curves, the other curves are not smooth. The possible reason is that when training the model, the data that can improve the performance of the model is selected from 10 times of processing data every time, which belongs to the local optimal value, so the selected data may not really improve the performance of the model.

4. Overall Implementation of the System

Based on the system requirements analysis and overall design, this section further analyzes and processes each module of speaker recognition, including data processing, speaker verification, speaker identification, speaker verification and experimental performance verification modules. In this section, the implementation method of each module is introduced in detail, and the implementation effect of the module is shown at the same time.

Use tkinter to generate all interfaces. The main interface of the system is shown in Fig.4.

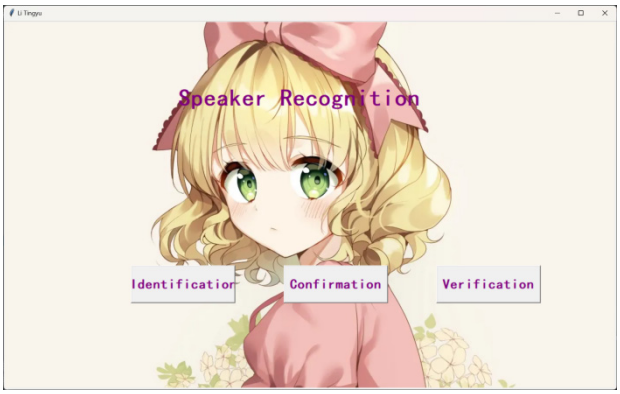

Fig 4. System main interface

4.1. Speaker Verification Module

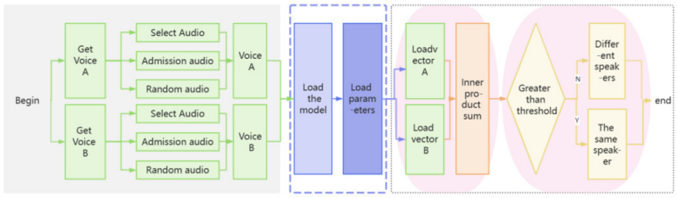

Fig 5. Speaker verification module

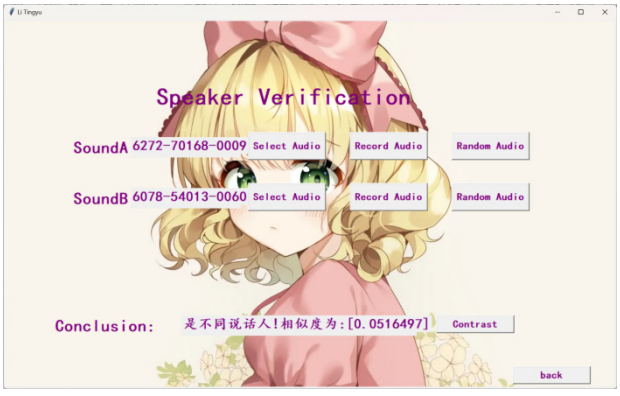

Fig 6. Speaker verification interface

ISSN: 2688-8645

This function can obtain audio in three ways: random audio, selected audio and recorded audio.

random audio is obtained by storing all tags in a list, using sample function in random library to randomly select a speaker, and then storing the tags of the speaker's speech text in the list, randomly selecting a speech of the speaker.

The selected audio is obtained by using the askopenfilename method in QFileDialog library to obtain the local file, and the absolute path of voice selection is each speech of each person in "D:\ Deep_Speaker-speaker_recognition_system-master\ Deep_Speaker-speaker_recognition_system-master\ audio\ LibriSpeechSamples\ train-clean-100", and the label of the speaker is obtained from the path.

When recording audio, pyaudio library is used to record audio. After recording audio, librosa package is called for denoising operation. The system does not know the speaker id, and displays "audio1" or "audio2" in the name box.

Three ways of acquiring audio can appear and be used together, for example: Sound A uses "selected audio", Sound B uses "random audio" and so on.

Click the comparison button, first create a model, load the optimal parameters, process the selected two speech data, and input them into the model to obtain the embedded vector of speech, and then calculate the inner product sum of the two embedded vectors. Compared with the threshold, if it is greater than the threshold (the threshold is 0.5), it will be the same speaker, otherwise, it will be different speakers. Finally, the judgment result and the similarity of the two sounds (the similarity range is in [-1, 1]) are output to the text box.

4.2. Speaker Identification and Confirmation Module

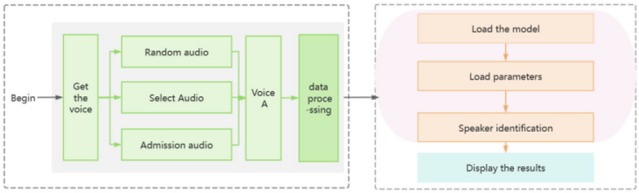

Fig 7. Speaker identification and confirmation module

4.2.1. Speaker Identification

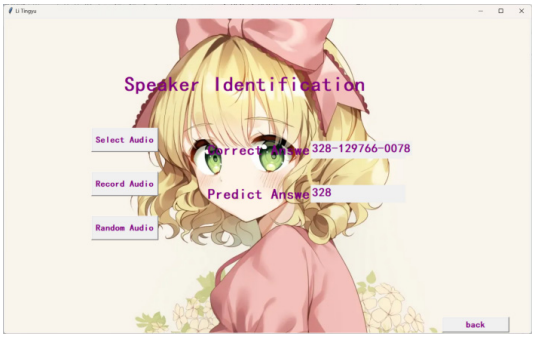

Fig 8. Speaker identification interface

Speaker recognition refers to the process of identifying which speaker in the data set is the voice to be recognized, that is, finding the target speaker among N people. In the speaker recognition interface, add buttons with different ways to obtain speech, and contain two text boxes, one to display the correct answer and the other to display the predicted answer. The page contains a "Return" button. Click the button to return to the main interface. Running this function, the speaker recognition running interface is shown in Figure Fig.8.There are also three ways to obtain the speech to be recognized: random audio, selective audio and recorded audio.

The button clicking event is associated with the function corresponding to the mode of acquiring voice. After acquiring voice, the acquired voice is stored as npy file through the data

ISSN: 2688-8645

processing module, and then the vector feature of the voice bar is formed. The voice vector of the nearest speaker is matched in the feature vector library of the stored database in advance, and the prediction result is displayed in the prediction answer text box, and the correct answer is displayed in the standard answer text box at the same time.

4.2.2. Speaker Confirmation

Speaker confirmation refers to specifying a speaker and judging whether the speaker's voice is the voice of the designated person. The method of acquiring speech is to select a speech mode, randomly select a speech from a data set, process the acquired speech by the same steps of data processing, extract the characteristics of speech bar vectors, and match with the characteristic vectors in the corpus to obtain the most likely speaker name, compare it with the randomly selected speaker, and output the comparison result.

In the speaker confirmation interface, it is also necessary to add a button to select the audio mode, select the speaker's name drop-down box, the real speaker information text box and the predicted answer text box. Click the Back button to return to the main interface. Running this function, the speaker confirmation interface is shown in Fig.9.

Usage: Randomly select a speaker from the left drop-down box, and then click the Select Audio button on the right to select a voice from the data set. The system automatically loads the model, processes the voice data, inputs the voice characteristics into the model for prediction, and obtains the recognition result, which is displayed on the interface.

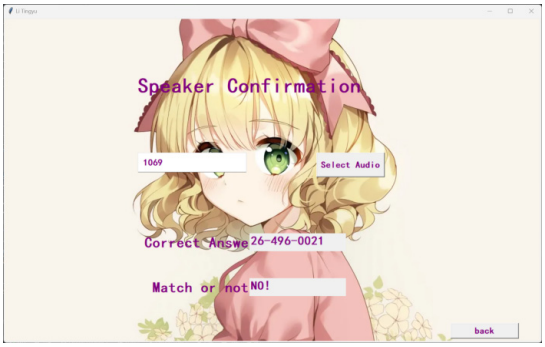

Fig 9. Speaker confirmation interface

5. Conclusion

In this practice, we build a speaker recognition system. The main processes include: selecting data sets, data preprocessing, model training, extracting embedded vectors from the trained models, and finally distinguishing different speakers by using vector feature library and verifying the identity of speakers.

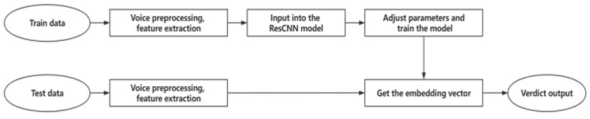

Fig 10. Overall flow

References

[1] Li C, Ma X, Jiang B ,et al. Deep Speaker: an End-to-End Neural Speaker Embedding System [J]. arXiv, 2017. DOI:10.48550/arXiv.1705.02304.

[2] Li, Mu, and Others. Dive into Deep Learning, 2020, http://d2l.ai/.

[3] He, Kaiming, et al. "Deep residual learning for image recognition." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.