Research on Intelligent Speech Interaction System Based on Residual Neural Network and Baidu Speech Platform

Qiongfei Wu, Junhao Wu, Yi Chen, Zhiqiang Zhang
School of Intelligent Engineering, Wuhan Institute of Design and Science, Wuhan, Hubei China

Abstract: To address the dual needs of convenience and security in human-computer interaction under the context of the Internet of Things (IoT) and Artificial Intelligence (AI), a system has been designed based on Raspberry Pi4B, which integrates voice recognition, speech synthesis, and speaker verification functions. Voice recognition and speech synthesis capabilities leverage Baidu's speech platform technology, while speaker verification employs a Residual Neural Network (ResNet34) model based on the PyTorch framework. With a focus on enhancing the user experience, the system incorporates the snowboy offline voice wake-up engine for voice interaction and utilizes Python's Tkinter library to implement a customized graphical user interface (GUI). After strict testing and verification, this system not only efficiently and friendly meets various voice interaction scenarios in the field of Internet of Things technology, but also utilizes voiceprint recognition technology to ensure the application security of the system. It also provides research value for the innovation of open-source hardware platforms in the field of artificial intelligence.

Keywords: Voice Recognition; Speech Synthesis; Speaker Verification; Baidu Voice; Neural Network

1. Introduction

In the context of the digital and intelligent era, the application range of voice interaction systems is increasingly extensive. From smart home control and personal voice assistants in daily life to customer service, medical health consultation, educational tutoring, and in-car navigation systems in professional fields, their influence has penetrated all aspects of society. Traditional voice interaction systems convert users' voice commands into computer-understandable instructions through voice recognition technology and then use speech synthesis technology to feedback the computer's processing results to users in the form of speech, thereby achieving natural interaction between humans and computers [1].

However, most of the similar voice interaction products on the market are based on cloud services, and issues of user privacy and data security are gradually gaining attention. Against this background, this project aims to explore how to integrate the most advanced voice technology and deep learning network models on low-cost hardware devices while considering the addition of voiceprint information recognition technology for identity verification and security. This realizes an intelligent interaction function that includes voice wake-up, voice recognition, speech synthesis, and voiceprint information recognition, promoting the development of low-cost, high-performance, convenient, and secure voice interaction terminal technology.

2. System Solution Design

The hardware part of the project uses Raspberry Pi 4B (8GB) as the main control, a USB driver-free microphone for audio input, and a 3.5mm adjustable volume speaker for audio output. The software part of the project design scheme is as follows:

(1) Voice Wake-up Module: Using the snowboy offline voice wake-up engine to achieve Raspberry Pi wake-up word listening, detecting the wake-up word to create or restore the voice function interface.

(2) Voice Recognition Function: Using Baidu Voice Recognition SDK to achieve real-time recognition of a single segment of speech not exceeding 60 seconds in any length and displaying the recognition results in the text box of the voice function interface.

(3) Speech Synthesis Function: Using Baidu Speech Synthesis SDK to select synthetic

audio with different tones for any length of target text for content playback.

(4) Voiceprint Information Recognition Function: Using the ResNet34 deep residual neural network model deployed and loaded with training models on Raspberry Pi 4B to achieve user voiceprint registration, recognition, and cancellation.

The overall functional structure of the system is shown in Figure 1.

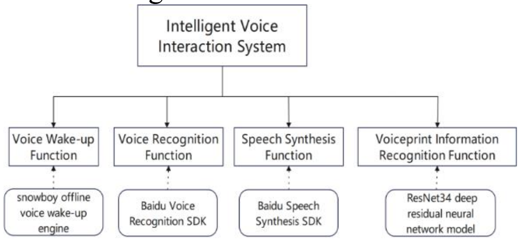

Intelligent Voice Interaction System
Voice Wake-up Function
Voice Recognition Function
Speech Synthesis Function
Voiceprint Information Recognition Function
snowboy offline voice wake-up engine
Baidu Voice Recognition SDK
Baidu Speech Synthesis SDK
ResNet34 deep residual neural network model

Figure 1. System Function Structure Diagram

3. Voice Wake-up

3.1 Snowboy Installation and Testing

Snowboy is a highly customizable wake word detection engine widely used in real-time embedded systems and always listening (even offline). Visit the Snowboy official website at https://snowboy.kitt.ai/, select a wake word you like from the "Hotword Library" (in this project, "snowboy" is selected), record your voice, and train your model. After recording and training the model, download the "snowboy. pmdl" file [2]. Then install the necessary dependencies:

Install PyAudio: sudo apt-get install python3-pyaudio.

Install SWIG: sudo apt-get install swig.

Install ATLAS: sudo apt-get install libatlas-base-dev.

Get the Snowboy source code: git clone https://github.com/Kitt-AI/snowboy. git

Extract and compile: cd snowboy/swig/Python3 && make

Test by running the built-in demo file (ensure the microphone and speaker are properly connected). When the wake word "snowboy" is triggered, the demo will play:

python demo.py snowboy.pmdl.

3.2 Voice Wake-up Software Design

The voice wake-up function is implemented through the start wake word listener function, which starts the wake word listener. When the wake word is detected, the detected callback callback function is called, which ends the current listener and calls the wake word detected function. This function determines whether the voice function interface is displayed; if so, it restores the window. If not, it displays the main window [3]. The system flowchart for this function is shown in Figure 2.

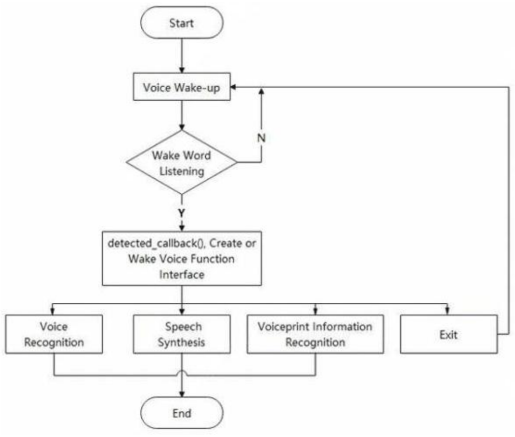
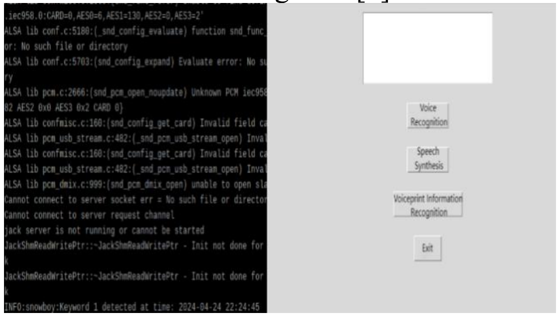

Start
Voice Wake-up
Wake Word Listening
N
Y
detected_callback(), Create or Wake Voice Function Interface
Voice Recognition
Speech Synthesis
Voiceprint Information Recognition
Exit
End

Figure 2. Voice Wake-up Flowchart

The voice wake-up GUI interface display function is shown in Figure 3 [4].



```
.iec958.0:CARD=0,AES0=6,AES1=130,AES2=0,AES3=2'
ALSA lib conf.c:5180:(_snd_config_evaluate) function snd_func_
or: No such file or directory
ALSA lib conf.c:5703:(snd_config_expand) Evaluate error: No su
ry
ALSA lib pcm.c:2666:(snd_pcm_open_noupdate) Unknown PCM iec958
82 AES2 0x0 AES3 0x2 CARD 0}
ALSA lib confmisc.c:160:(snd_config_get_card) Invalid field ca
ALSA lib pcm_usb_stream.c:482:(_snd_pcm_usb_stream_open) Inval
ALSA lib confmisc.c:160:(snd_config_get_card) Invalid field ca
ALSA lib pcm_usb_stream.c:482:(_snd_pcm_usb_stream_open) Inval
ALSA lib pcm_dmix.c:999:(snd_pcm_dmix_open) unable to open sla
Cannot connect to server socket err = No such file or director
Cannot connect to server request channel
jack server is not running or cannot be started
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for
k
JackShmReadWritePtr::~JackShmReadWritePtr - Init not done for
k
INFO:snowboy:Keyword 1 detected at time: 2024-04-24 22:24:45
```

Figure 3. Voice Wake-up GUI Display

4. Voice Recognition and Speech Synthesis

4.1 Voice Recognition Software Design

The voice recognition function is implemented through the toggle recording function. This function completes the recording of the audio to be recognized through two clicks of the voice recognition button (the first click is to record, the second click is to stop). The self.is recording flag in the MainWindow class is initialized to False.

(1) When the voice recognition button is pressed, the state of the voice recognition button changes to "Start Recording." The function checks whether the self.is recording flag in the MainWindow class is False; if it is, it calls the start recording function to start recording. The self.is recording flag changes to True, and the state of the voice recognition

button changes to "Stop Recording." If it is not, it calls the stop recording function to stop recording and clears any previous audio frame data to prevent interference with the voice recognition result. The system captures audio data from the microphone through PyAudio and stores it in the frame list by frames.

(2) When the voice recognition button is clicked again, recording stops. The accumulated audio frames during recording are written into a temporary .wav file. A VoiceRecognitionWorker thread is created and activated to asynchronously call the Baidu voice recognition service to recognize the temporary .wav file. A timer is set to check the recognition result queue every 100 milliseconds [5].

(3) The check recognition result function is called to check the recognition result queue. When the recognition result is not empty, the result is retrieved from the queue and the clear textbox function is called to clear the contents of the text box. The recognition result is then displayed in the text box of the voice function interface, and the temporary .wav file is deleted. The voice recognition button returns to the "Voice Recognition" state, awaiting the next recording operation.

The system flowchart for the voice recognition function is shown in Figure 4.

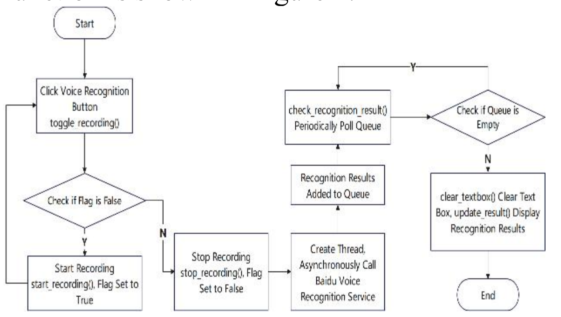

Start
Click Voice Recognition Button toggle recording()
Check if Flag is False
Y
N
Start Recording start_recording(), Flag Set to True
Stop Recording stop_recording(), Flag Set to False
Create Thread, Asynchronously Call Baidu Voice Recognition Service
Recognition Results Added to Queue
check_recognition_result() Periodically Poll Queue
Check if Queue is Empty
Y
N
clear_textbox() Clear Text Box, update_result() Display Recognition Results
End

Figure 4. Voice Recognition Function Flowchart

4.2 Speech Synthesis Software Design

The speech synthesis function is implemented through the on sv clicked function. When the speech synthesis button is clicked, the voice function interface is hidden, and the interface jumps to the speech synthesis GUI.

(1) Enter the target text content in the text box of the speech synthesis interface to facilitate the speech synthesis operation.

(2) Select the tone to be synthesized from the tone selection dropdown menu.

(3) When the "Select File Location" button is clicked, the on select file function is called to change the save path of the synthesized audio file.

(4) When the "Convert and Play" button is clicked, the on convert click function is called to check for any unfinished speech synthesis tasks. If there are none, the target text entered by the user in the text box and the save path of the synthesized audio file set by the user are retrieved. The text is divided into paragraphs not exceeding the maximum length. A thread pool is used to develop audio files for each paragraph. The successfully generated audio files are collected, the sampling rate is adjusted, and they are stitched into a complete audio file. The stitched audio is written to the specified output path [6]. PyGame is used to play the generated audio file.

(5) When the "Exit" button is clicked, the on exit click function is called to destroy the speech synthesis GUI and restore the voice function interface GUI.

The system flowchart for the speech synthesis function is shown in Figure 5.

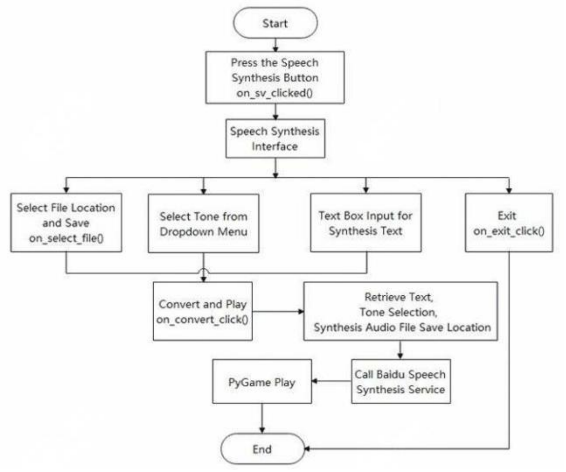

Start
Press the Speech Synthesis Button on_sv_clicked()
Speech Synthesis Interface
Select File Location and Save on_select_file()
Select Tone from Dropdown Menu
Text Box Input for Synthesis Text
Exit on_exit_click()
Convert and Play on_convert_click()
Retrieve Text, Tone Selection, Synthesis Audio File Save Location
Call Baidu Speech Synthesis Service
PyGame Play
End

Figure 5. Speech Synthesis Function Flowchart

5. Voiceprint Information Recognition

5.1 Voiceprint Recognition Model Theory

5.1.1 ResNet34 network model

The ResNet network, fully known as Residual Network, also called the residual neural network, is a new type of deep learning network model proposed by He Kaiming et al. in 2015, different from traditional deep convolutional neural networks. This model introduces the residual learning mechanism (i.e., residual blocks) to solve the network degradation phenomenon and the gradient vanishing/exploding problem in deep learning networks. Each residual block contains two paths: one is a shortcut connection that directly passes the input to the output, and the other transforms the input through several convolutional layers. The output of the residual block is the sum of these two paths, as shown in Figure 6 [7].

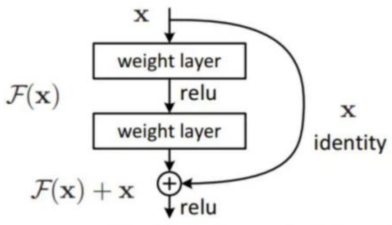

x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

Figure 6. Residual Block

The voiceprint recognition network model in this system adopts the ResNet34 network model, which is divided into two parts [8]:

(1) Basic components

IRBlock, a custom residual block, adds batch normalization processing and calls the PReLU activation function after each convolutional layer compared to the traditional residual block of ResNet. The IRBlock contains two convolutional layers, with the first not changing the input size and the second possibly having a stride for downsampling. Optionally, it includes an SEBlock (attention module).

(2) Main class

The first few layers of the network include an initial 3x3 convolutional layer, a max-pooling layer, and four stages of residual blocks (layer1 to layer4). The output channels for each stage are 64, 128, 256, and 512, respectively, with downsampling before each subsequent stage using a stride-2 convolution. The final structure reduces the feature map size to 1x1 using an adaptive max-pooling layer, followed by batch normalization, a Dropout layer to reduce overfitting, flattening through a Flatten layer, and a fully connected layer (fc5) that maps to a 512-dimensional feature space, followed by another batch normalization processing [9].

5.1.2 Model training

The voiceprint dataset used for training the ResNet34 network in this system is the open-source audio dataset zhvoice, containing about 3,200 speakers, with a total audio duration of about 900 hours, approximately 1.13 million actual speech texts, and about 13 million words. The audio data in the dataset undergoes format conversion, random cropping, splicing, short-time Fourier transform, and other data preprocessing operations to expand the dataset, remove invalid data, and generate a training and test set in a 4:1 ratio for ResNet34 network model training.

The model is trained on a PC with GPU power, modifying the ResNet34 input layer to match the dimensions of audio features (ResNet34 was originally used for image classification but can be adjusted for audio feature shapes, adapting the one-dimensional Mel-Frequency Cepstral Coefficients (MFCC) sequence to a suitable model input shape) [10]. A fully connected layer is added at the end of the ResNet34 model, with the number of output nodes equal to the number of categories (i.e., different speakers), using pretrained ResNet34 weights for initialization [11].

5.1.3 Application deployment

After multiple rounds of training, the saved ResNet34 network model is evaluated and screened. The model with the best recognition effect and excellent performance is selected for practical application [12]. Using a microphone to collect audio, the ResNet34 network model extracts the audio voiceprint features for comparison, finally deploying it to run on the Raspberry Pi 4B. The system runs and executes the voiceprint recognition identity verification function for testing to check if the functionality is achieved.

5.2 Voiceprint Recognition Software Design

Voiceprint information recognition and verification are implemented through the on ss clicked function. When the voiceprint recognition identity verification button is clicked, it switches to the voiceprint recognition identity verification GUI, hiding the voice function interface GUI. The audio library and voiceprint data are loaded for the first time, and the voiceprint recognition interface text box displays in real time. The overall flowchart for this function is shown in Figure 7.

When the voiceprint recognition button is clicked, the on recognition clicked function is called to create and activate an instance of the RecordAudioThread recording thread class, recording thread, to record audio and generate the audio file to be recognized. The handle recording result function is called to load the registered voiceprint data, and the recognition function is called to perform recognition [13]. The function extracts the voiceprint feature vector from the audio file to be recognized and normalizes it. The cosine similarity between the normalized feature vector and the existing normalized voiceprint feature vectors is calculated and compared with the recognition threshold to make a judgment, returning the user name and cosine similarity, and popping up a recognition result window. The flowchart for this function is shown in Figure 8 [14,15].

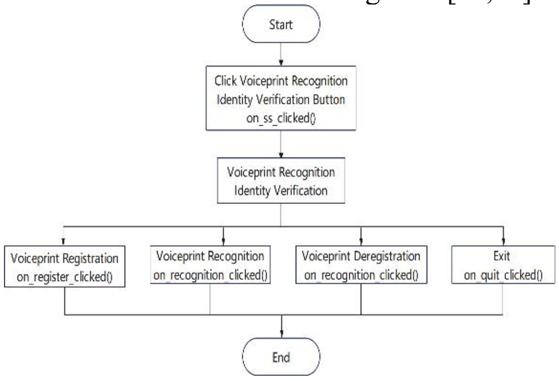

Start
Click Voiceprint Recognition Identity Verification Button on_ss_clicked()
Voiceprint Recognition Identity Verification
Voiceprint Registration on_register_clicked()
Voiceprint Recognition on_recognition_clicked()
Voiceprint Deregistration on_recognition_clicked()
Exit on_quit_clicked()
End

Figure 7. Voiceprint Recognition GUI Flowchart

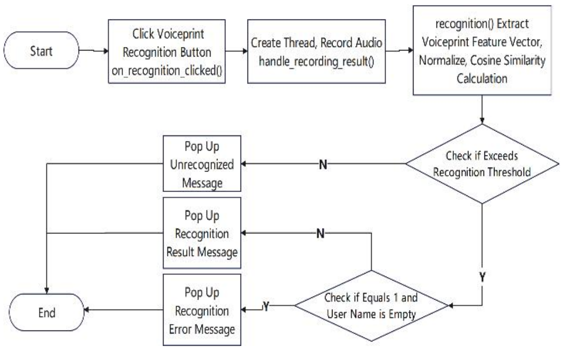

Start
Click Voiceprint Recognition Button on_recognition_clicked()
Create Thread, Record Audio handle_recording_result()
recognition() Extract Voiceprint Feature Vector, Normalize, Cosine Similarity Calculation
Check if Exceeds Recognition Threshold
N
Pop Up Unrecognized Message
Y
Check if Equals 1 and User Name is Empty
N
Pop Up Recognition Result Message
Y
Pop Up Recognition Error Message
End

Figure 8. Voiceprint Recognition Function Flowchart

6. Conclusion

This system studied the simulated implementation of an intelligent voice interaction system based on Raspberry Pi. Using Raspberry Pi 4B, a USB driver-free microphone, and a 3.5mm adjustable volume speaker, a simple intelligent voice interaction system based on Raspberry Pi was built. The Baidu Voice Development Platform was used to implement voice recognition and speech synthesis functions. Innovatively, the ResNet34 neural network model was introduced to address the security issues of voiceprint identity verification in the voice interaction process. Users can set the playback content and tone style of the synthesized speech according to their needs, satisfying the customization service requirements of voice synthesis through this user-initiated selection mode. The GUI interaction program was designed in a customized way, using a voice wake-up passive trigger graphical interface interaction method. The entire system operation logic is concise, the operation process is simple and convenient, while also ensuring the privacy of user information and the security and reliability of system use.

References

[1] He Song, Huang Wei, Wu Xiyao, et al. Design of Intelligent Voice Interaction Robot Based on Cloud Platform. Software Engineering, 2021-24 (04): 55-59.

[2] Lu Man, Chen Jiayue. Design of Intelligent Voice Interaction System Based on Raspberry Pi. Technology and Innovation, 2023, (18): 47-49.

[3] Huang Yiqiu, Zhou Cheng, et al. Design of Intelligent Household Medicine Box Control System. Internet of Things Technology, 2019, 9 (07): 95-96+100.

[4] Li Gang Python: Instance based GUI (Tkinter) programming. Computer programming skills and maintenance, 2022, (06): 7-9.

[5] Liu Shilong, Xie Dian, Tang Zhiyuan, et al. Voice interaction method for intelligent guided cane. Internet of Things Technology, 2024, 14 (03): 128-130.

[6] Sun Lin, Yang Lin. Research on the Application of Speech Technology Teaching Based on Baidu AI Platform. China Education Technology Equipment, 2021, (15): 117-118+126.

[7] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 770-778.

[8] Xing Xiaohai. Research on Voiceprint Recognition Based on Residual Networks and Attention Mechanisms. Ningxia University, 2022.

[9] Li Yuxin, Wang Jiaxin, Liu Lijun. Garbage classification recognition mini program based on ResNet34 convolutional neural network. Computer and Information Technology, 2024, 32 (02):1-3.

[10] Zhang Hailong, Wang Liheng, Ji Xinran. Design of a deep learning based voiceprint

recognition identity verification system. Automation and Instrumentation, 2024, 39 (04):130-134.
[11] Yeyupiaoling. (2022). Voiceprint recognition system based on Pytorch. https://github.com/yeyupiaoling/VoiceprintRecognition-Pytorch.
[12] Bajrami X, Gashi B. Face recognition with Raspberry Pi using deep neural networks. International Journal of Computational Vision and Robotics, 2022, 12(2):177-193.
[13] Bi Zongying. Research on Voiceprint Recognition Model Based on Deep Learning. Central North University, 2023.
[14] Yanxiong Li, Zhongjie Jiang. Speaker verification using attentive multi-scale convolutional recurrent network. Applied Soft Computing, Volume 126, 2022, 109291, ISSN 1568-4946.
[15] Ali Bou Nassif, Ismail Shahin. Speech Recognition Using Deep Neural Networks: A Systematic Review. IEEE Access, vol. 7, pp. 19143-19165, 2019.