4.1.6 IP+Voice broadcast

1) Definition of socket

Socket is an intermediate software abstraction layer for communication between the application layer and the TCP/IP protocol family. It is a set of interfaces.

2) Workflow of socket

First, the server initializes the Socket. Then, binds to the port, listens to the port, calls accept to block, and waits for the client to connect.
At this time, if a client initializes a Socket and then connects to the server, if the connection successfully, the client and the server establish a connection.
The client sends a data request, the server receives the request and deal with this request.
Next, sends the response data to the client, the client reads the data. Finally, closes the connection.
The above is a complete interaction.

3)Using of Socket function

```
import socket    # Import socket library
socket.socket(socket_family,socket_type,protocal=0)
#socket_family can be AF_UNIX or AF_INET. socket_type can be SOCK_STREAM or SOCK_DGRAM. protocol is generally not filled in, and the default value is 0.

#Get tcp/ip socket
tcpSock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

#Get udp/ip socket
udpSock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

#Because there are too many attributes in the socket module. We used the 'from module import *' statement here. Using 'from socket import *', we will bring all the attributes in the socket module to our namespace, which can greatly shorten our code.
#Eg: tcpSock = socket(AF_INET, SOCK_STREAM)
```

- **Server socket function**

```
s.bind ()    #Bind (host, port number) to the socket
s.listen ()    #Start TCP listening
s.accept () #Passively accept TCP client connections, (blocking) waiting for the connection
```

- **Client socket function**

```
s.connect()        #Initializes the TCP server connection
s.connect_ex()    #connect() an extended version of the function that returns an error code
```

- **Socket functions for public use**

s.recv() # Receive TCP data
s.send() # Send TCP data (s.send() When the data waiting to be sent is greater than the remaining space in the own buffer area, the data is lost and will not be sent)
s.sendall() # Send complete TCP data (essentially, s.send() is called cyclically, s.sendall() the data waiting to be sent is greater than the remaining space in the own buffer area. Data will not lose data, s.send() is called cyclically until finished.

s.recvfrom() # Receive UDP data
s.sendto() # Send UDP data
s.getpeername() # Connect to the remote address of the current socket
s.getsockname() # The address of the current socket
s.getsockopt() # Returns the parameter for the specified socket
s.setsockopt() # Sets the parameters of the specified socket
s.close() # Close socket

- **Lock-oriented socket method**

s.setblocking() #Set the blocking and non-blocking mode of the socket
s.settimeout() #Set the timeout for blocking socket operations
s.gettimeout() #Get the timeout time for blocking socket operations

- **File-oriented socket methods**

s.fileno() #The file descriptor of the socket
s.makefile() #Create a file related to the socket

In the above function, we used s.getsockname () #The address of the current socket

The following code returns your own connection information when connecting to the Baidu webpage. Therefore, we only need to read the first index value of the array to be the local IP address of the Raspblock of the Raspberry Pi, and combine the speech synthesis tutorial in the previous chapter to broadcast the obtained IP address.

Code path
/home/pi/Yahboom_Project/4.AI Voice course/ 05.IP_Voice.ipynb

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.connect(('www.baidu.com', 0))
print(s.getsockname())
ip = s.getsockname()[0]
print(ip)
```

Result:

```
import socket
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.connect(('www.baidu.com', 0))
print(s.getsockname())
ip = s.getsockname()[0]
print(ip)
```

```
('192.168.1.93', 41395)
192.168.1.93
```

Voice broadcast local IP address:

```
#!/usr/bin/env python3

# for get ip
import socket, time
import pygame
from aip import AipSpeech

""" Voice technology APPID AK SK """
SpeechAPP_ID = '17852430'
SpeechAPI_KEY ='eGeO4iQGAjHCrzBTYd1uvTtf'
SpeechSECRET_KEY = 'Cn1EVsUngZDbRLv4OxAFrDHSo8PsvFVP'

Speechclient = AipSpeech(SpeechAPP_ID, SpeechAPI_KEY, SpeechSECRET_KEY)

# Obtain IP address
def getip():
    try:
        s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
        s.connect(('www.baidu.com', 0))
        ip = s.getsockname()[0]
    except:
        ip = "x.x.x.x"
    finally:
        s.close()
    return ip

if __name__ == "__main__":
    while(1):
        try:
            ip = getip()
            print(ip)
            if(ip == "x.x.x.x"):
                continue
            if(ip != "x.x.x.x"):
                #result = Speechclient.synthesis(text = ip, 'zh', 1,
```

```
options={'spd':3,'vol':9,'per':2,})
                        result = Speechclient.synthesis("The ip is " + ip, 'zh', 1, {'spd':3,
'vol' : 1, 'per' : 3} )

                        #Write the synthesized speech into a file
                        if not isinstance(result,dict):
                                with open('audio.mp3','wb') as f:
                                        f.write(result)
                                        # We can use Raspberry Pi pygame
                                        pygame.mixer.init()
                                        pygame.mixer.music.load('./audio.mp3')
                                        pygame.mixer.music.play()
                                        time.sleep(10)
                                        pygame.mixer.init()
                                        pygame.mixer.music.load('./audio.mp3')
                                        pygame.mixer.music.play()
                                        time.sleep(10)
                                        break
                        else:
                                print(result)
                except Exception:
                        raise
```