

山东师范大学

《人机交互技术》

期末报告

题 目： 针对缺失模态的无障碍人机对话系统

学生姓名： 隋 远

学 号： 201911010105

专 业： 计算机科学与技术(非师范)

指导教师： 杜 萍

学 院： 信息科学与工程学院

2021 年 12 月 15 日

目录

一、项目简介	1
1.1 背景介绍	1
1.2 项目介绍	1
二、用户需求	2
2.1 视觉障碍用户需求	2
2.2 听力障碍用户需求	3
2.3 运动控制能力障碍用户需求	3
三、任务分析	3
3.1 信息无障碍的主要科学问题	3
3.2 无障碍人机对话系统架构	4
四、功能模块	5
4.1 输入模块	5
4.2 实时语音识别模块	5
4.3 语音合成模块	6
4.4 图像识别模块	6
4.5 眼动识别模块	6
4.6 人机会话机器人	7
五、界面设计	7
5.1 界面风格定位	7
5.2 界面迭代设计	7
5.3 客户端界面设计	7
5.4 仿微信端 UI 界面设计	19
六、功能实现	22
七、可用性与用户体验评价	34
7.1 测试流程	34
7.2 测试结果与分析	37
附录:	37

一、项目简介

1.1 背景介绍

信息无障碍(information accessibility)是一个学科交叉的技术和应用领域，旨在用信息技术弥补残障人士生理和认知能力的不足，让他们可以顺畅地与他人、物理世界和信息设备进行交互。据中国残联统计，中国现有8500万残疾人，是世界上残疾人口最多的国家。其中，听力残疾2000万人，视力残疾1200万人，各类肢体残疾2500万人，智力残疾和精神残疾1200万人.....随着社会老龄化程度加重，残疾人口数量也在持续增长。互联网和用户终端的普及，使得信息无障碍成为一个越来越值得关注的领域，目标是解决残障人士的信息访问甚至是生活服务问题。

1.2 项目介绍

本项目借助百度开放平台，搭建一套面向缺失模态人群的信息无障碍人机交互系统。主要实现的功能有图像识别（动物识别、植物识别、果蔬识别、文字识别、通用物品及场景识别等）、语音识别与合成（短语音识别、短语音合成）、眼动识别、人机对话（包含问候、机器人控制以及屏幕控制等模块）。

产品用例图如下图所示：其中图像识别、语音识别、人机对话分别含有子模块用于实现不同细分功能；人机对话对应的机器人控制采用自适应学习方式，可以针对用户进行迭代学习，不断提升精度。

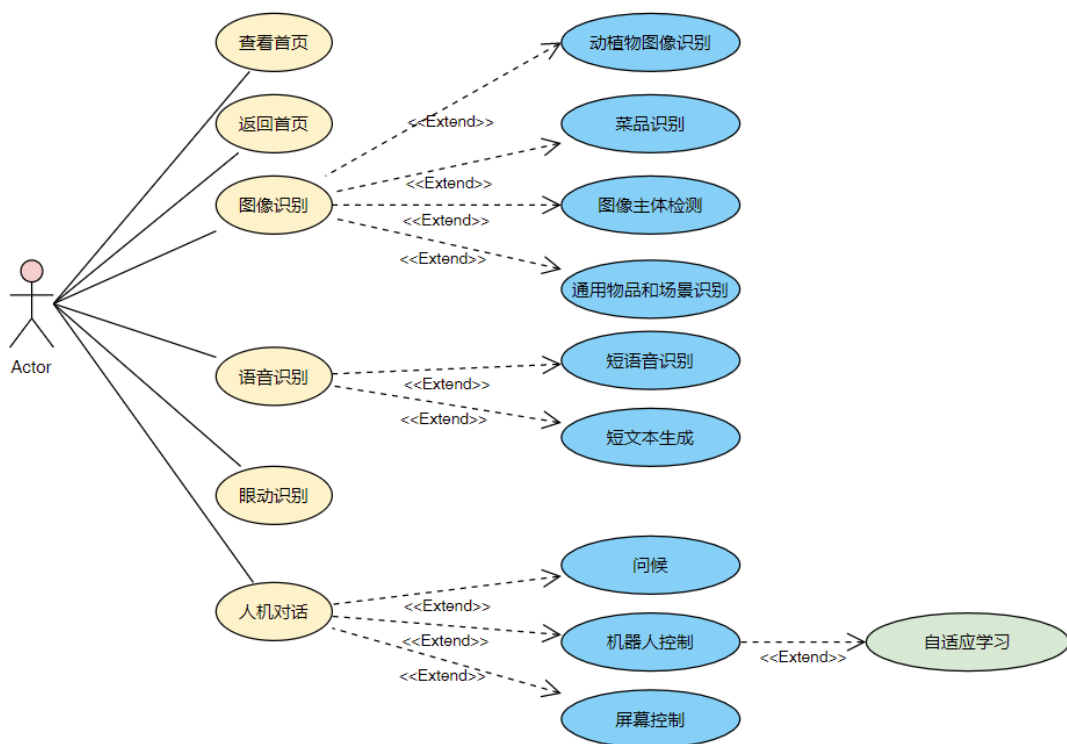


图 1 无障碍人机对话系统用例图

二、用户需求

残障类型多样，用户需要的无障碍技术也不尽相同，本项目面向三类主要的残障类型（视觉障碍、听觉障碍和运动障碍）人群遇到的问题和主要的技术方案。

2.1 视觉障碍用户需求

视力残疾用户的需求包括**独立出行、识别身边物体、与信息设备交互等**。针对独立出行的需求，目前有基于计算机视觉的道路识别技术，通过立体声场或者震动反馈为视力残疾用户指示方向。但是这些设备目前还不能取代盲杖，还需要更多的技术突破。针对识别物体的需求，主要是利用视频/图像转换为文本的技术，例如 Seeing AI、谷歌的 Lookout 以及读屏程序 VoiceOver、Talkback，可以通过语音读出获得焦点的控件信息，这样视力残疾用户通过听就能了解设备界面上的信息内容。

针对视力障碍用户，本项目后端采用图像识别技术，将静态/动态图

片信息转换为用户可识别接受的信息传递方式（如文字信号或语音信号），实现视频/图像转换为文本/语音，从而满足视力障碍用户需求。

2.2 听力障碍用户需求

听力残疾用户面临的主要问题是与人交流存在障碍，以及观看视频内容时听不到声音。老年听力障碍是指随着年龄增长，听觉器官的衰老和退变所导致的听觉功能下降，发病率居世界第三位。助听器设备通过放大声音信号，可解决“听不到”的问题；但对于听觉中枢受损的人，声音信号分析能力却难以弥补，解决不了“听得清”的问题。

针对听力障碍用户，本项目后端采用语音识别技术，将音频信号实时转换为文字信号或其他听障用户可以接受的信息传递方式，从而满足听障用户需求。

2.3 运动控制能力障碍用户需求

运动控制能力缺失的用户，包括上肢残疾，或者患帕金森症、脑瘫、肌肉萎缩、渐冻症等疾病的用户。他们丧失了灵活控制手指运动的能力，而手指是人表达交互意图的主要运动器官，也是电脑和手机的主要操作器官。在构建面向这类用户的信息无障碍交互技术时，其中一个难题是用户的差异性，几乎每个用户的可运动部位及其运动能力都是不同的，给构建适合于个体的通用输入技术带来了挑战。

针对视力障碍用户，本项目后端采用眼动识别技术，将用户“眼动”作为输入方式，但值得注意的是，由于眼动识别通常难以完成“确认”操作，容易产生误触发，且操作精度有限。

三、任务分析

3.1 信息无障碍的主要科学问题

音视频的理解和信息转换（主要针对听障和视障）。视觉和听觉是人们接受信息的主要感官。听障和视障用户因为缺乏某种感官而无法完整理解信息，需要建立音视频的理解技术，用机器算法理解音视频内容的语义，进而转换为用户可用感官能接受的信息类型，包括音频和文字之间的语音

识别和文本到语言(Text To Speech, TTS)技术, 图像到文字和视频到文字的技术。目前, 精度是主要问题, 尤其是克服多种噪声条件下的高精度实现, 对于这些技术的可用性起到关键作用。

图形用户界面到声音界面的编码转换(主要针对视障)。个人电脑和手机都是图形用户界面, 信息以可视的方式传递给用户, 而视障用户只能通过听觉(触觉为辅)来接收信息, 相比于视觉, 不仅信息接收的带宽要低很多, 而且信息呈现的模式也发生了变化。视觉提供整体和并行的信息获取能力, 听觉只能提供局部串行的信息。这也会影响用户对于交互界面的心理模型, 进而影响到交互决策。因此, 需要研究从图形界面到声音界面的编码转换方法, 优化“读屏”的方法。

个性化信息输入和意图理解(主要针对视障和运动障碍)。人体的运动控制系统包括运动执行和反馈两部分。运动障碍用户无法精确灵活地控制手指运动, 视障用户由于缺少视觉反馈也不能做精确的输入控制, 导致物理运动自由度受限和运动控制精度低的问题。前者需要开发具有个性化能力的输入技术, 根据用户实际可以控制的输入范围来映射有效的输入; 后者需要实现从有噪声的运动控制数据中提取用户的交互意图。

3.2 无障碍人机对话系统架构

信息无障碍是以用户为中心的交互方案, 是对人的交互性能的优化。优秀的信息无障碍技术要适应用户的生理和认知能力, 而不是让用户适应技术。为此, 要采用智能交互方法来开展研究, 从用户角度来设计和创新适用的交互模式, 通过智能传感、智能用户意图推理和智能信息呈现来构建信息无障碍的交互界面。

本系统主要包含两大部分: 一部分针对不同人群需求, 设计不同智能交互识别模块, 用于完成不同模态信息之间的转换; 另一部分搭建人机对话系统, 系统可以根据用户指令完成一些列操作并给予一定的智能反馈。针对系统设计, 分别画出识别模块时序图及无障碍人机对话时序图如下图所示。

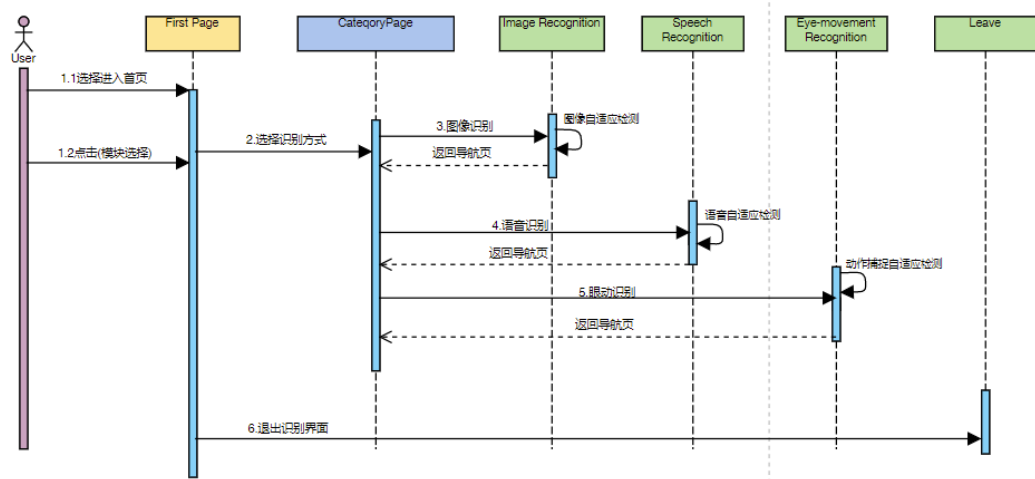


图 2 识别模块时序图

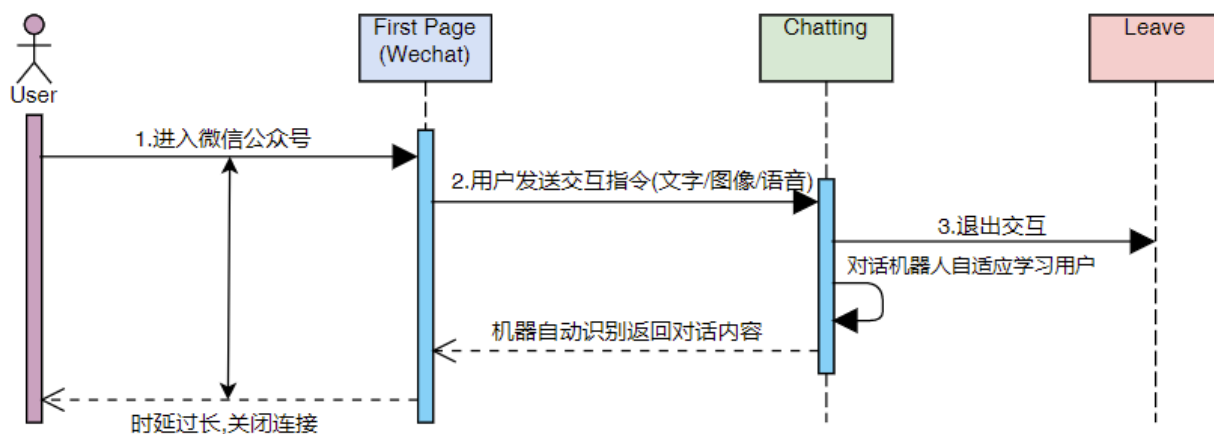


图 3 无障碍人机对话时序图

四、功能模块

4.1 输入模块

针对不同的用户需求，需要设计不同的输入模块，如针对听障和视障人群，播放音视频时用户因缺乏某种感官而导致无法完整理解信息，需要建立音视频的理解技术，此时输入模块应当根据用户缺乏模态信息，将特征输入机器学习算法理解音视频语义，进而转换为用户可用感官能接受的信息类型。

4.2 实时语音识别模块

通过语音识别实现人机对话，解决视障用户需求。将语音对话实时识别为文字，实现自然流畅的人机对话。

输入：输入接口同时接受两种输入格式：(a) 实时音频流输入，要求上传实时，不能过快，即整体耗时略多于原始音频流，若因为网络不稳定不过导致需要重新发起请求续传的，允许超发一段 XXms 的音频，待网络恢复后将全部音频传给服务端。(b) 音频文件输入，支持 pcm 格式的音频文件，每 160ms 为一帧发送，间隔 1-2ms，整体耗时短于音频流输入。

处理：调用百度开放平台实时语音识别 websocket API 接口，进行实时语音识别，并将转换得到的文字信号做进一步文字解析后，传递给语音生成模块。

4.3 语音合成模块

通过语音合成将机器生成的文字信号转换为语音信号传递给视障用户，完成信息交互。

输入：输入接口接受各类文本格式文件，使用 UTF-8 编码，要求文本长度必须小于 1024 字节；若文本较长，可以采用多次请求的方式。

处理：调用百度开放平台在线语音合成 SDK 接口，获得语音合成能力。

输出：输出指定音频格式的语音信号，借助播放设备播放语音，完成流程的人机交互。

4.4 图像识别模块

针对失声障碍人群，通过图像识别模块，识别手语信息，并转换为文字信号，实现人机交互。

输入：输入接口接受手语图像，要求图片可正常解码、长宽比例合适，背景相对模糊。

处理：借助百度开放平台搭建手语识别模块，对常见的手语信号进行文字转换。

输出：输出手语信息对应的文本信号，完成人机交互。

4.5 眼动识别模块

针对运动控制能力障碍人群，通过眼动识别模块，实现与机器的交互操作。

输入：输入接口利用移动设备摄像头捕捉用户眼动信号

处理：通过系统内嵌眼动识别模块，对用户眼动信号进行分析，推断用户眼动信息，从而实现与机器进行交互。

4.6 人机会话机器人

针对用户的不同模态输入，自动识别用户需求，返回用户需要的信息。

五、界面设计

5.1 界面风格定位

首先根据缺失模态人群的特性，确定系统的风格以及主色调。选取紫色调为系统的主要风格，整体效果的表现形式以间接明快风格为主，与交互式按钮相结合，充分体现对于缺失模态人群的关注和理解。

5.2 界面迭代设计

针对缺失模态的无障碍人机对话系统，为了达到可用性和用户体验目标，采用迭代设计方法，即在整个设计的过程中，重复“设计-评估-再设计-再评估”的过程，在反复的实践过程中最大程度地满足用户的需求。

下面以导航页面进行说明本系统如何采用迭代设计方法：设计初始阶段，拟采用三个模块叠放的方式进行页面导引，如图 1 所示，但此版本界面并没有体现出系统面向无障碍人群的特色。因此对导航页进行版本迭代如图 2 所示，在原有模块内容不变的基础上，添加一个麦克风按钮（可进行语音识别）自动识别用户需求，很好适配视障人士需求。

5.3 客户端界面设计

下面具体介绍界面的详细设计，由于系统界面一致性原则，主要针对系统中关键代表性模块的界面设计进行说明。

(1) 首页:用户可通过点按的方式或语音输入的方式，选择各个功能模

块，如图像识别、语音识别以及眼动识别。

1. 语音识别（麦克风）：用户可以通过直接语音输入的方式选择想要的功能模块，如用户可以说出“我想识别一张图片”，系统将识别用户需求自动点按图像识别模块，切换到图像识别模块。

2. 图像识别功能模块：用户可以通过拍照或上传图片的方式识别图像信息（支持动物识别、植物识别、果蔬识别、图像主体识别、通用物品和场景识别）。

3. 语音识别功能模块：用户可以通过上传一段音频信号或实时说一段指令识别音频信号，也可以输入一段文字返回对应文字的一段音频信号（支持短语音识别和短语音合成）。

4. 眼动识别功能模块：用户可以根据眼动进行交互指令（例如翻页）



图 4 导航页面迭代版本 1



图 5 导航页面迭代版本 2

(2) 图片识别模块：用户可以通过下方的点击添加图片，添加图片，机器将同时返回语音信息和文字信息。

通过点击添加图片（或拍照识别的方式），实现动物、植物、果蔬、文字的识别。系统将自动根据图像内容返回图片传递的信息（以语音信息

和文字信息的方式返回)。

1. 识别文字结果:



图 6 文字识别结果

2. 识别动物结果:



图 7 动物识别结果

3. 识别植物结果:



图 8 植物识别结果



图 9 图像识别模块+人机对话

(3) 眼动识别模块：用户通过点按下方按钮，点击后可以立即对准镜头，此时机器将根据用户眼动信号完成一系列指令（如翻页等功能）；针

对视觉障碍人群，也可以通过语音输入的方式初始化机器，进行一系列指令的输入。



图 10 眼动识别模块

(4) 语音识别模块：用户通过点按下方按钮，按住语音说话，机器自

动将识别到的音频信号转换为文字信号并输出到显示屏幕上，完成人机对话。



图 11 语音识别模块+人机对话

(5) 屏读模块：针对视障用户，采用图像识别模块对屏幕上图像进行读取转换为语音信号传递给用户。用户可以在读屏设置界面，选择是否开启读屏模式（也可以选择人声、语速等）



图 12 读屏设置

(6) 无障碍人机对话协议：在系统打开界面，应当弹出软件相关协议，标注系统对于用户信息的保密及安全性保证，由于面向用户为残障人群，需要充分注重用户隐私。



图 13 无障碍人机对话协议

5.4 仿微信端 UI 界面设计

1. 人机对话：屏幕控制，用户可以通过输入屏幕操作指令，系统可以根据用户命令进行一系列的交互行为。



图 14 人机对话：屏幕控制

2. 人机对话：机器人控制，用户可以通过输入机器人控制指令，系统可以根据用户命令进行一系列控制机器人操作的交互行为。



图 15 人机对话：机器人控制

3. 人机对话：问候。用户可以对系统发出一些问候语言，系统将自动识别语言类型并返回相对应的问候语言。



图 16 人机对话：问候

六、功能实现

本系统开发基于百度开放平台实现。

1. 语音识别模块：

百度短语音识别可以将 60 秒以下的音频识别为文字。适用于语音对话、语音控制、语音输入等场景。

接口类型：通过 REST API 的方式提供的通用的 HTTP 接口。适用于任意操作系统，任意编程语言

接口限制：需要上传完整的录音文件，录音文件时长不超过 60 秒。浏览器由于无法跨域请求百度语音服务器的域名，因此无法直接调用 API 接口。

支持音频格式：pcm、wav、amr、m4a

音频编码要求：采样率 16000、8000，16 bit 位深，单声道

调用流程：

1. 创建账号及应用：在 ai.baidu.com 控制台中，创建应用，勾选开通 “语音技术” “-” 短语音识别、短语音识别极速版 “能力。获取 AppID、API Key、Secret Key，并通过请求鉴权接口换取 token，详见 “接入指南”。
2. 创建识别请求：POST 方式，音频可通过 JSON 和 RAW 两种方式提交。JSON 方式音频数据由于 base64 编码，数据会增大 1/3。其他填写具体请求参数，详见 “请求说明”。
3. 短语音识别请求地址：http://vop.baidu.com/server_api
4. 返回识别结果：识别结果会即刻返回，采用 JSON 格式封装，如果识别成功，识别结果放在 JSON 的 “result” 字段中，统一采用 utf-8 方式编码。详见 “返回说明”。

1. 从网页中申请的应用获取 appKey 和 appSecret

```
API_KEY = '4QSHUs3o7KPGV6UWqw3fKYQf'
SECRET_KEY = 'C4BMG8ngBNcKzzBSkwVRluZZQ2kkQIND'
```

2. 设置要识别的文件，文件格式

```
# 需要识别的文件
AUDIO_FILE = './audio/16k.pcm' # 只支持 pcm/wav/amr 格式，极速版额外支持m4a 格式
# 文件格式
FORMAT = AUDIO_FILE[-3:] # 文件后缀只支持 pcm/wav/amr 格式，极速版额外支持m4a 格式
```

3. 设置采样率


```
# 采样率
RATE = 16000 # 固定值
```

4. JSON 格式 POST 上传本地文件

```
params = {'dev_pid': DEV_PID,
          #"lm_id" : LM_ID,      #测试自训练平台开启此项
          'format': FORMAT,
          'rate': RATE,
          'token': token,
          'cuid': CUID,
          'channel': 1,
          'speech': speech,
          'len': length
        }
post_data = json.dumps(params, sort_keys=False)
# print post_data
req = Request(ASR_URL, post_data.encode('utf-8'))
req.add_header('Content-Type', 'application/json')
```

5. 返回结果

两种上传方式都返回统一的结果，采用 JSON 格式封装，如果识别成功，识别结果放在 JSON 的 “result” 字段中，统一采用 utf-8 方式编码。

字段名	数据类型	必需	描述
err_no	int	必填	错误码
err_msg	string	必填	错误码描述
sn	string	必填	语音数据唯一标识，系统内部产生。如果反馈及 debug 请提供 sn。

若识别成功返回 case

```
{"err_no":0,"err_msg":"success.", "corpus_no":"15984125203285346378", "sn":"481D633F-73BA-726F-49EF-8659ACCC2F3D"}
```

若识别失败返回 case

```
{"err_no":2000,"err_msg":"data empty.", "sn":"481D633F-73BA-726F-49EF-8659ACCC2F3D"}
```

6. 测试结果

```
audio_voice_assistant_get
SUCCESS WITH TOKEN: 24.25fc81293069e73bb1ff74856dcb9d01.2592000.1638930969.282335-25040199 EXPIRES IN SECONDS: 2592000
Request time cost 1.581294
{"corpus_no":"7028022362544274389","err_msg":"success.", "err_no":0, "result":["北京科技馆。"], "sn":"231044343771636338970"}
```

正确识别音频信号“北京科技馆”。

2. 语音合成模块：

1. 安装语音合成 Python SDK

2. 新建 AipSpeech (语音合成的 Python SDK 客户端)

```
from aip import AipSpeech
```

```
""" 你的 APPID AK SK """
```

```
APP_ID = '你的 App ID'
```

```
API_KEY = '你的 Api Key'
```

```
SECRET_KEY = '你的 Secret Key'
```

```
client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
```

3. 调用 API 接口, 实现语音合成

```
result = client.synthesis('你好百度', 'zh', 1, {
    'vol': 5,
})

# 识别正确返回语音二进制 错误则返回dict 参照下面错误码
if not isinstance(result, dict):
    with open('audio.mp3', 'wb') as f:
        f.write(result)
```

4. 优化代码实现读取 txt 文件批量语音合成

```
from aip import AipSpeech

APP_ID = '25035344'
API_KEY = 'PB0z3aK8mepgUG0d8HjL7kUG'
SECRET_KEY = 'QHgPTGDPjFy2KLQ0o8bK2fUvAFhYE0s'

def aipSpeech(str_id):
    client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
    result = client.synthesis(str_id, 'zh', 1, {'spd': 6, 'vol': 5, 'per': 106})
    filename = 'result/audio'
    if not isinstance(result, dict):
        with open(filename + '{}.mp3'.format(id), 'wb') as f:
            f.write(result)
    text = []

if __name__ == '__main__':
    file = open('text.txt', encoding='utf-8')
    while True:
        lines = file.readlines(5000)
        if not lines:
            break
        for line in lines:
            text.append(line)
    for i in range(len(text)):
        aipSpeech(text[i])
```


1. 测试样本数据如下：


“你好百度


今天天气如何

我想去济南逛一逛,你有什么推荐吗”

生成样本音频文件如下：

 audio1.mp3

 audio2.mp3

 audio3.mp3

2. API 可调整音频音调、语速、音量、人声等

参数	类型	描述	是否必须
tex	String	合成的文本, 使用UTF-8编码, 请注意文本长度必须小于1024字节	是
cuid	String	用户唯一标识, 用来区分用户, 填写机器 MAC 地址或 IMEI 码, 长度为60以内	否
spd	String	语速, 取值0-9, 默认为5中语速	否
pit	String	音调, 取值0-9, 默认为5中语调	否
vol	String	音量, 取值0-15, 默认为5中音量	否
per	String	发音人选择, 0为女声, 1为男声, 3为情感合成-度逍遥, 4为情感合成-度丫丫, 默认为普通女	否

3. 图像识别模块：

1. 读取图像

```
def get_file_content(self, filePath):  
    with open(filePath, 'rb') as fp:  
        return fp.read()
```

从./images 中读取 example.jpg 文件

2. 请求百度 AipImageClassify 权限

```
""" 你的 APPID AK SK """  
APP_ID = '25211913'  
API_KEY = 'H01HC6LxH564BWG6UcK8TDVv'  
SECRET_KEY = 'S0gAUjj6LIY4LizaBctD06F8s4iDf0U7'  
client_img = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)
```

3. 调用 API

```
def advance_general_reg(self):
    options = {}
    options["baike_num"] = 5 # 返回百科信息的结果数
    return client_img.advancedGeneral(self.img, options)
```

4. 返回百度百科信息

```
def get_information(self):
    ret = self.advance_general_reg()
    # print(ret["result"][1]["keyword"])
    # 图片描述，返回百科描述及关键词（作为文件命名）
    return ret["result"][1]["baike_info"]["description"], ret["result"][1]["keyword"]
```

猫，属于猫科动物，分家猫、野猫，是全世界家庭中较为广泛的宠物。家猫的祖先据推测是古埃及的沙漠猫，波斯的波斯猫，已经被人类驯化了3500年（但未像狗一

实际操作过程中，设定五种图像识别方式如下：

```
def get_img_rec(file, schema):
```

```
    image = get_file_content(file)
```

```
    if schema == "通用识别":
```

```
        """ 如果有可选参数 """
```

```
        options = {}
```

```
        options["baike_num"] = 5
```

```
        """ 带参数调用通用物体和场景识别 """
```

```
        result = client.advancedGeneral(image, options)["result"]
```

```
        keyword = result[1]["keyword"]
```

```
        baike_url = result[1]["baike_info"]["baike_url"]
```

```
        image_url = result[1]["baike_info"]["image_url"]
```

```
        description = result[1]["baike_info"]["description"]
```

```
        info = "图像主体类别: "+keyword +"\n"+" 百度百科 url: "+baike_url+"\n"+\n
```

```
            "百度百科图片_url: "+image_url+"\n"+"描述: "+description
```

```
        return info
```

```
    elif schema == "动物识别":
```

```
        """ 如果有可选参数 """
```

```

options = {}
options["top_num"] = 3
options["baike_num"] = 5

""" 带参数调用动物识别 """
result = client.animalDetect(image, options)["result"]
keyword = result[1]["name"]
score = result[1]["score"]
info = "动物名称: " + keyword + "\n" + "置信度: "+score
return info
elif schema == "植物识别":
    """ 如果有可选参数 """
    options = {}
    options["baike_num"] = 5

    """ 带参数调用植物识别 """
    result = client.plantDetect(image, options)["result"]
    keyword = result[1]["name"]
    score = str(result[1]["score"])
    info = "植物名称: " + keyword + "置信度: " + score
    return info
elif schema == "红酒识别":
    result = client.redwine(image)["result"]["wineNameCn"]
    return result
elif schema == "货币识别":
    result = client.currency(image)["result"]
    cur_name = result["currencyName"]
    cur_domin = result["currencyDenomination"]

    info = "货币种类: "+cur_name+"\n"+"货币面值: "+cur_domin
    return info

```

4. 文字识别模块:

1. 获得 access-token:

```
def get_access_token():
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
          'd3505rqkfipyzDKqRBr7oCp&client_secret=3hWQ736DLq20xh8pybnww2GknWtZzueB'
    # host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
    #       'eZfr4d4wDu1Ike3Fw4KFobuF&client_secret=TEIvgHjjVRRPEtqbG9B9XtK4UzQY3wLj'
    response = requests.get(host)
    if response:
        return response.json()["access_token"]
```

2. 调用 API 接口, 识别图像文字

```
import requests
import base64

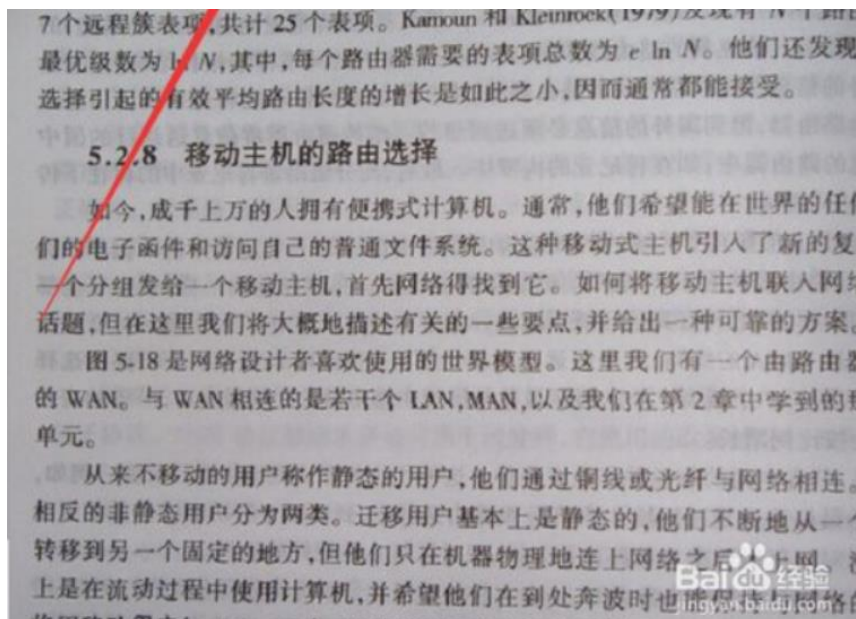
'''
通用文字识别（高精度版）
'''

def get_access_token():
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
          'd3505rqkfipyzDKqRBr7oCp&client_secret=3hWQ736DLq20xh8pybnww2GknWtZzueB'
    # host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
    #       'eZfr4d4wDu1Ike3Fw4KFobuF&client_secret=TEIvgHjjVRRPEtqbG9B9XtK4UzQY3wLj'
    response = requests.get(host)
    if response:
        return response.json()["access_token"]

request_url = "https://aip.baidubce.com/rest/2.0/ocr/v1/accurate_basic"
# 二进制方式打开图片文件
f = open('../images/img2txt.png', 'rb')
img = base64.b64encode(f.read())

params = {"image":img}
access_token = get_access_token()
print(access_token)
request_url = request_url + "?access_token=" + access_token
headers = {'content-type': 'application/x-www-form-urlencoded'}
response = requests.post(request_url, data=params, headers=headers)
if response:
    print(response.json())
```

3. 测试



选择一张测试图片,测试文字识别是否准确。

返回结果如下：

{'words_result': [{'words': '7 个远程簇表项. 共计 25 个表项。Kamoun 和 Kleinrock(1979)发现有 N 个路由'}, {'words': '最优级数为 LN, 其中, 每个路由器需要的表项总数为 $e \ln N$ 。他们还发现'}, {'words': '选择引起的有效平均路由长度的增长是如此之小, 因而通常都能接受。'}, {'words': '5.28 移动主机的路由选择'}, {'words': '今, 成千上万的人拥有便携式计算机。通常, 他们希望能在世界的任何'}, {'words': '们的电子函件和访问自己的普通文件系统。这种移动式主机引入了新的复'}, {'words': '个分组发给一个移动主机, 首先网络得找到它。如何将移动主机联入网络'}, {'words': '话题, 但在这里我们将大概地描述有关的一些要点, 并给出一种可靠的方案。'}, {'words': '图 5-18 是网络设计者喜欢使用的世界模型。这里我们有一个由路由器'}, {'words': '的 WAN。与 WAN 相连的是若干个 LAN, MAN, 以及我们在第 2 章中学到的那'}, {'words': '单元。'}, {'words': '感从来不移动的用户称作静态的用户, 他们通过铜线或光纤与网络相连。'}, {'words': '相反的非静态用户分为两类。迁移用户基本上是静态的, 他们不断地从一个'}, {'words': '转移到另一个固定的地方, 但他们只在机器物理地连上网络之后网漫'}, {'words': 'Baldu 经部'}, {'words': '上是在流动过程中使用计算机, 并希望他们在到处奔波时也能持同络的'}, {'words': '田致动田白(。'}]}, 'words_result_num': 17, 'log_id': 1469265997381223084}

```
D:\ProgramData\Anaconda3\python.exe D:/workSpace/Human-Machine/AipSpeech/textRecognition.py
24.80d3e44e7fb8cc1285cc003dbb0f2520.2592000.1641727285.282335-25326221
<Response [200]>
{'words_result': [{'words': '7个远程簇表项. 共计25个表项。Kamoun和Kleinrock(1979)发现有N个路由'}, {'words': '最优级数为LN, 其中, 每个路由器需要的表项总数为 e ln N。他们还发现'}, {'words': '选择引起的有效平均路由长度的增长是如此之小, 因而通常都能接受。'}, {'words': '5.28 移动主机的路由选择'}, {'words': '今, 成千上万的人拥有便携式计算机。通常, 他们希望能在世界的任何'}, {'words': '们的电子函件和访问自己的普通文件系统。这种移动式主机引入了新的复'}, {'words': '个分组发给一个移动主机, 首先网络得找到它。如何将移动主机联入网络'}, {'words': '话题, 但在这里我们将大概地描述有关的一些要点, 并给出一种可靠的方案。'}, {'words': '图 5-18 是网络设计者喜欢使用的世界模型。这里我们有一个由路由器'}, {'words': '的 WAN。与 WAN 相连的是若干个 LAN, MAN, 以及我们在第 2 章中学到的那'}, {'words': '单元。'}, {'words': '感从来不移动的用户称作静态的用户, 他们通过铜线或光纤与网络相连。'}, {'words': '相反的非静态用户分为两类。迁移用户基本上是静态的, 他们不断地从一个'}, {'words': '转移到另一个固定的地方, 但他们只在机器物理地连上网络之后网漫'}, {'words': 'Baldu 经部'}, {'words': '上是在流动过程中使用计算机, 并希望他们在到处奔波时也能持同络的'}, {'words': '田致动田白(。'}]}, 'words_result_num': 17, 'log_id': 1469265997381223084}
Process finished with exit code 0
```

5. 人机对话机器人

1. 设定 APPID、AK、SK

```
""" 你的 APPID AK SK """
APP_ID = '25211913'
API_KEY = 'H01HCGlxH564BWG6UcK8TDVv'
SECRET_KEY = 'S0gAUjjGLIY4LizaBctD0GF8s4iDf0U7'

client_img = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)
client_speech = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
```

2. 将对话机器人封装成一个类 Aip

```
client_img = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)
client_speech = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
```

```
class Aip:
    def __init__(self, filePath):
        self.img = self.get_file_content(filePath)
```

```
"""返回access_token"""
```

```
def get_access_token(self):
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
          'eZfr4d4wDu1Ike3Fw4KFobuF&client_secret=TEIvgHjjVRRPEtqbG9B9XtK4UzQY3wLj'
    response = requests.get(host)
    if response:
        return response.json()["access_token"]
```

```
"""测试对话机器人"""
```

```
def test_access(self):
    url = 'https://aip.baidubce.com/rpc/2.0/unit/service/v3/chat?access_token=' + self.get_access_token()
    # post_data = {"version":3.0,"service_id":"S61779","session_id":["1127231,1127232,1127233"],
    # "log_id":"7758521","request":{"terminal_id":"88888","query":"你好a"}}
    post_data = '{"version":3.0,"service_id":"S61779","skill_id":["1127231,1127232,1127233"],' \
                '"session_id":["",""],"log_id":"7758521","request":{"terminal_id":"88888","query":"你好a"}}'
    headers = {'content-type': 'application/x-www-form-urlencoded'}
    response = requests.post(url, data=post_data.encode("utf-8"), headers=headers)
    if response:
        print(response.json()["result"]["responses"][0]["actions"][0]["say"])
        return response.json()["result"]["responses"][0]["actions"][0]["say"]
```

3. main 函数调用 ()

```
if __name__ == '__main__':
    filePath = "./images"
    aip = Aip(filePath + "/example.jpg")
    aip.get_information()
    # aip.speech_synthesis()
    # aip.test_access()
```

6. 部分 GUI 界面设计

使用 EasyGUI 完成部分的界面设计：

其核心代码如下：

```
flag = True
while flag:
    choice = g.choicebox(msg="请选择系统功能", title="无障碍人机对话系统", choices=["图像识别", "文字识别", "语音识别", "人机对话"])
    if choice == "图像识别":
        choice = g.choicebox(msg="请选择需要识别图像的类别", title="图像识别", choices=["动物识别", "植物识别", "红酒识别", "货币识别", "通用识别"])
        # 读取图片
        img_path = g.fileopenbox(msg="请选择需要识别的图像", title="图像识别", filetypes=["*.png", "*.jpg", "*.jpeg"])
        if img_path is not None:
            img_rec = g.msgbox(image=img_path, msg=get_img_rec(img_path, choice), title="图像识别", ok_button="确定并返回")
            if not g.boolbox(msg="是否继续进行智能交互?"):
                flag = False
    elif choice == "文字识别":
        # 读取文字图片
        img_path = g.fileopenbox(msg="请选择需要识别的图像", title="文字识别", filetypes=["*.png", "*.jpg", "*.jpeg"])
        if img_path is not None:
            txt_rec = g.msgbox(image=img_path, msg=get_text_rec(img_path), title="文字识别", ok_button="确定并返回")
            if not g.boolbox(msg="是否继续进行智能交互?"):
                flag = False
    elif choice == "语音识别":
        # 读取语音
        audio_path = g.fileopenbox(msg="请上传需要识别(转换)的录音", title="语音识别", filetypes=["*.pcm", "*.wav", "*.amr"])
        if audio_path is not None:
            audio_rec = g.msgbox(msg=get_speech_rec(audio_path), title="语音识别", ok_button="确定并返回")
            if not g.boolbox(msg="是否继续进行智能交互?"):
                flag = False
```

Demo 实现：

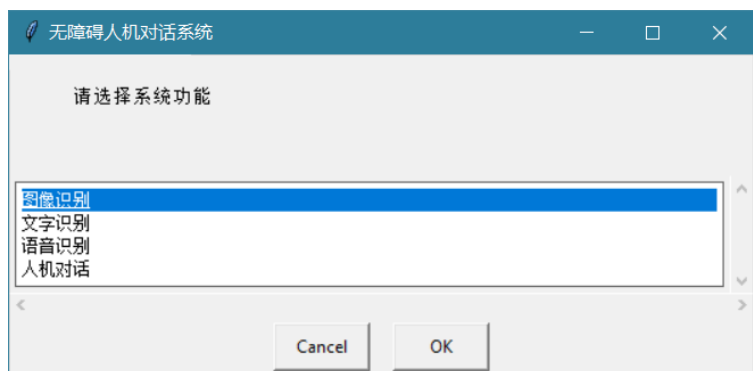


图 17 demo 示例 1

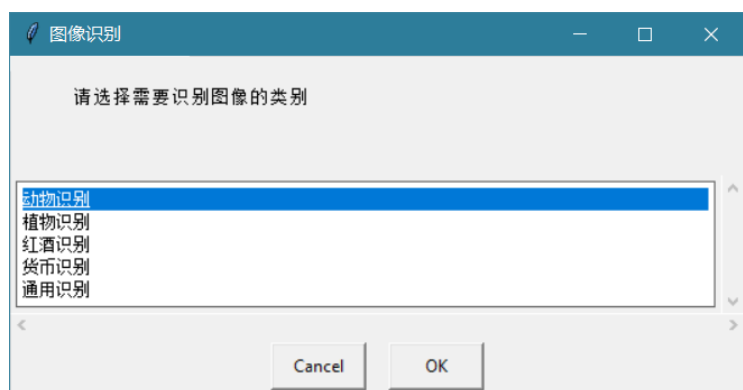


图 18 demo 示例 2 图像识别类别



图 19 demo 示例 3 纸币识别



图 20 demo 示例 4 红酒识别



图 21 demo 示例 5 动物识别



图 22 demo 示例 6 植物识别

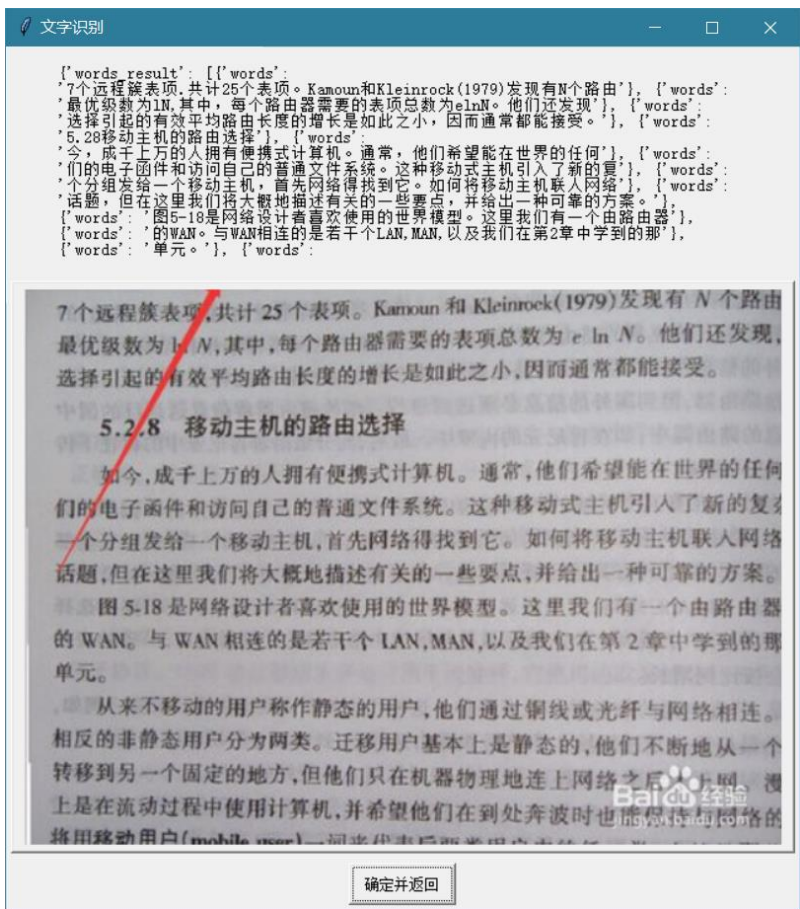


图 23 demo 示例 7 文字识别

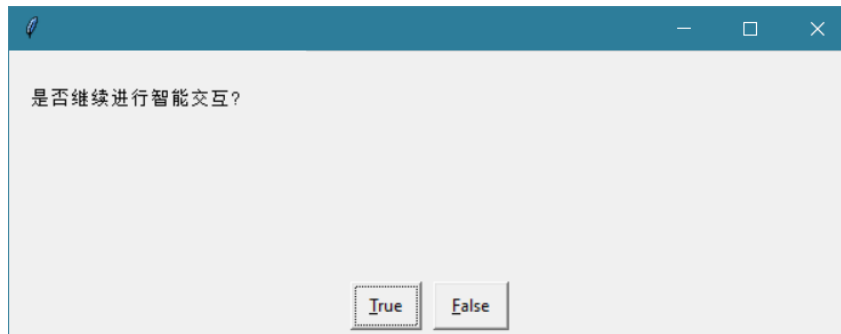


图 24 demo 示例 8

七、可用性与用户体验评价

7.1 测试流程

1. 设计测试：参与者：舍友 3 名；调整好实验设备。
2. 测试准备：运行一个示范性测试（针对模型的各个模块进行一次示范性测试，在执行可用性测试之前，检测测试方案是否存在问题以及任务是否可行），指导被试者进行练习后，正式开始测试
3. 测试阶段：-测试前：让参与者填写测试前调查问卷；编写脚本；-测试过程中：为每一项任务维护一个日志；构建一个问题列表，问题列表中的内容是没有出现在核对表中的；标注某些问题，并写下这些问题发生的前提条件；让被试者填写一份用户交互满意度调查问卷，从而分析系统的可用性。

无障碍人机对话系统交互满意度调查问卷

感谢您能抽出几分钟时间来参加本次答题，现在我们就马上开始吧！

1.您的年级是？ *

- ☐ 大一
- ☐ 大二
- ☐ 大三

2.您觉得人机对话软件应该做到什么？ *

- ☐ 服务器可同时容纳大量用户
- ☐ 保密性好
- ☐ 稳定
- ☐ 无广告
- ☐ 操作简单易懂
- ☐ 其他（请补充）

3.在使用人机对话时您遇到过哪些问题？ *

- ☐ 网络拥堵，客户不能及时回复
- ☐ 人机对话语言重复，缺乏拟人

4.您认为无障碍人机对话系统相对其他人机对话平台的“优势”？ *

- ☐ 简单易操作
- ☐ 能够很好适用于残障人群
- ☐ 体现了对缺失模态人群的关照
- ☐ 其他（请补充）

5.您觉得无障碍人机对话平台对残障人士的生活有什么帮助？ *

- ☐ 为生活提供了便利
- ☐ 提高了效率
- ☐ 提升了自学能力
- ☐ 使生活更自由、更方便
- ☐ 没有帮助

6.您认为使用无障碍人机交互时，其上手及后续操作友好度如何？ *

- ☐ 非常方便
- ☐ 比较方便
- ☐ 一般
- ☐ 比较不便
- ☐ 非常不便

7.您知道无障碍人机交互的所有功能，并应用灵活 *

- ☐ 是
- ☐ 不是

8.请您根据实际情况选择以下内容的符合程度 *

	非常不满意				非常满意
	1	2	3	4	5
系统可以为缺失模态人群提供方便，更好地生活	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
系统的各个功能均可以正常运行，且操作流畅	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
系统用户可以方便的进行反馈意见	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>
系统既可以采用APP终端使用，也可以在微信公众号端使用	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

9.您希望无障碍人机对话系统增加什么功能？ *

提交

图 25 用户满意度调查问卷

4. 处理数据：

- 收集数据（收集出错数及原因，观测核对表、问题列表等）
- 总结数据（总结测试过程中对某些特定事件印象以及自己的想法、参与者存在问题列表）

-组织材料（为数据建立表格归档、组织整理调查问卷和问题列表、备份音视频材料等）

7.2 测试结果与分析

根据无障碍人机对话系统交互满意度调查问卷结果可知，用户对系统的功能性和意义十分满意，总的评价为系统上手方便、使用灵活、对于残障人士可以提供一定程度的无障碍服务。但同时对于部分功能提出了更高的要求，比如人机对话系统时常会出现文不对题的现象，即系统有时会出现无法准确识别用户意图的情况。针对这一情况，本系统在迭代版本中采用一种自适应学习方法，针对用户新的输入进行学习，系统将自动适应用户输入习惯，从而返回更智能、更合理的回复。

附录：

1. 语音合成代码：

```
from aip import AipSpeech

APP_ID = '25035344'
API_KEY = 'PBOz3aK8mepgUGOdBHjL7kUG'
SECRET_KEY = 'QHgPTGPDPjFy2KLQ0o8bK2fUvAFhYE0s'

def aipSpeech(str, id):
    client = AipSpeech(APP_ID, API_KEY, SECRET_KEY)
    result = client.synthesis(str, 'zh', 1, {'spd':6, 'vol': 5, 'per': 106})
    filename = 'result/audio'
    if not isinstance(result, dict):
        with open(filename + '{}'.format(id) + '.mp3', 'wb') as f:
            f.write(result)
    text = []

if __name__ == '__main__':
    file = open('text.txt', encoding='utf-8')
```

```

while True:
    lines = file.readlines(5000)
    if not lines:
        break
    for line in lines:
        text.append(line)
    for i in range(len(text)):
        aipSpeech(text[i], i+1)
2. 语音识别
# coding=utf-8

import sys
import json
import base64
import time

IS_PY3 = sys.version_info.major == 3

if IS_PY3:
    from urllib.request import urlopen
    from urllib.request import Request
    from urllib.error import URLError
    from urllib.parse import urlencode
    timer = time.perf_counter
else:
    from urllib2 import urlopen
    from urllib2 import Request
    from urllib2 import URLError
    from urllib import urlencode
    if sys.platform == "win32":
        timer = time.clock
    else:
        # On most other platforms the best timer is time.time()

```



```

timer = time.time

# API_KEY = 'kVcnfD9iW2XVZSMaLMrtLYIz'
# SECRET_KEY = 'O9o1O213UgG5LFn0bDGNtoRN3VWl2du6'
API_KEY = '4QSHUs3o7KPGV6UWqw3fKYQf'
SECRET_KEY = 'C4BMG8ngBNcKzzBSkwVRluZZQ2kkQIND'

# 需要识别的文件
AUDIO_FILE = './audio/16k.pcm' # 只支持 pcm/wav/amr 格式，极速版额外支持 m4a
格式
# 文件格式
FORMAT = AUDIO_FILE[-3:] # 文件后缀只支持 pcm/wav/amr 格式，极速版额外支
持 m4a 格式

CUID = '123456PYTHON'
# 采样率
RATE = 16000 # 固定值

# 普通版

DEV_PID = 1537 # 1537 表示识别普通话，使用输入法模型。根据文档填写 PID，选
择语言及识别模型
ASR_URL = 'http://vop.baidu.com/server_api'
SCOPE = 'audio_voice_assistant_get' # 有此 scope 表示有 asr 能力，没有请在网页里勾
选，非常旧的应用可能没有

#测试自训练平台需要打开以下信息， 自训练平台模型上线后，您会看见 第二步：“”
获取专属模型参数 pid:8001， modelid:1234”，按照这个信息获取 dev_pid=8001，
lm_id=1234
# DEV_PID = 8001 ;
# LM_ID = 1234 ;

# 极速版 打开注释的话请填写自己申请的 appkey appSecret ，并在网页中开通极速版

```

（开通后可能会收费）

```
# DEV_PID = 80001
# ASR_URL = 'http://vop.baidu.com/pro_api'
# SCOPE = 'brain_enhanced_asr' # 有此 scope 表示有极速版能力，没有请在网页里开
通极速版
```

```
# 忽略 scope 检查，非常旧的应用可能没有
# SCOPE = False
```

```
class DemoError(Exception):
    pass
```

```
"""  TOKEN start """
```

```
TOKEN_URL = 'http://openapi.baidu.com/oauth/2.0/token'
```

```
def fetch_token():
    params = {'grant_type': 'client_credentials',
              'client_id': API_KEY,
              'client_secret': SECRET_KEY}
    post_data = urlencode(params)
    if (IS_PY3):
        post_data = post_data.encode( 'utf-8')
    req = Request(TOKEN_URL, post_data)
    try:
        f = urlopen(req)
        result_str = f.read()
    except URLError as err:
        print('token http response http code : ' + str(err.code))
        result_str = err.read()
```

```

if (IS_PY3):
    result_str = result_str.decode()

print(result_str)
result = json.loads(result_str)
print(result)
if ('access_token' in result.keys() and 'scope' in result.keys()):
    print(SCOPE)
    if SCOPE and (not SCOPE in result['scope'].split(' ')): # SCOPE = False 忽略检
查
        raise DemoError('scope is not correct')
    print('SUCCESS WITH TOKEN: %s EXPIRES IN SECONDS: %s' %
(result['access_token'], result['expires_in']))
    return result['access_token']
else:
    raise DemoError('MAYBE API_KEY or SECRET_KEY not correct: access_token
or scope not found in token response')

""" TOKEN end """

if __name__ == '__main__':
    token = fetch_token()

    speech_data = []
    with open(AUDIO_FILE, 'rb') as speech_file:
        speech_data = speech_file.read()

    length = len(speech_data)
    if length == 0:
        raise DemoError('file %s length read 0 bytes' % AUDIO_FILE)
    speech = base64.b64encode(speech_data)
    if (IS_PY3):
        speech = str(speech, 'utf-8')

```

```

params = {'dev_pid': DEV_PID,
          #"lm_id" : LM_ID,      #测试自训练平台开启此项
          'format': FORMAT,
          'rate': RATE,
          'token': token,
          'cuid': CUID,
          'channel': 1,
          'speech': speech,
          'len': length
          }

post_data = json.dumps(params, sort_keys=False)
# print post_data
req = Request(ASR_URL, post_data.encode('utf-8'))
req.add_header('Content-Type', 'application/json')
try:
    begin = timer()
    f = urlopen(req)
    result_str = f.read()
    print ("Request time cost %f" % (timer() - begin))
except URLError as err:
    print('asr http response http code : ' + str(err.code))
    result_str = err.read()

if (IS_PY3):
    result_str = str(result_str, 'utf-8')
print(result_str)
with open("result.txt", "w") as of:
    of.write(result_str)

```

3. 图像识别

```
from aip import AipImageClassify
```

```
""" 你的 APPID AK SK """
```

```
APP_ID = '23642440'
```

```

API_KEY = 'hbY0dSexZzqhgrwzehkbjnLT'
SECRET_KEY = 'oSX8LeRe4ZWIXryMEXc9Nzf8Yk7lUriA'

client = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)

""" 读取图片 """
def get_file_content(filePath):
    with open(filePath, 'rb') as fp:
        return fp.read()

def get_img_rec(file,schema):
    image = get_file_content(file)

    if schema == "通用识别":
        """ 如果有可选参数 """
        options = {}
        options["baike_num"] = 5

        """ 带参数调用通用物体和场景识别 """
        result = client.advancedGeneral(image,options)["result"]
        keyword = result[1]["keyword"]
        baike_url = result[1]["baike_info"]["baike_url"]
        image_url = result[1]["baike_info"]["image_url"]
        description = result[1]["baike_info"]["description"]

        info = "图像主体类别: "+keyword+"\n"+"百度百科 url: "+baike_url+"\n"+\
            "百度百科图片_url: "+image_url+"\n"+"描述: "+description
        return info

    elif schema == "动物识别":
        """ 如果有可选参数 """
        options = {}
        options["top_num"] = 3
        options["baike_num"] = 5

```

```

""" 带参数调用动物识别 """
result = client.animalDetect(image, options)["result"]
keyword = result[1]["name"]
score = result[1]["score"]
info = "动物名称: " + keyword + "\n" + "置信度: "+score
return info

elif schema == "植物识别":
    """ 如果有可选参数 """
    options = {}
    options["baike_num"] = 5

    """ 带参数调用植物识别 """
    result = client.plantDetect(image, options)["result"]
    keyword = result[1]["name"]
    score = str(result[1]["score"])
    info = "植物名称: " + keyword + "置信度: " + score
    return info

elif schema == "红酒识别":
    result = client.redwine(image)["result"]["wineNameCn"]
    return result

elif schema == "货币识别":
    result = client.currency(image)["result"]
    cur_name = result["currencyName"]
    cur_domin = result["currencyDenomination"]

    info = "货币种类: "+cur_name+"\n"+"货币面值: "+cur_domin
    return info

# get_img_rec("images/cash.png","通用识别")
4. 文字识别:
# encoding:utf-8

```

```

import requests
import base64

'''
通用文字识别（高精度版）
'''

def get_access_token():
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
    'd3505rqkfipyzDKqpRBr7oCp&client_secret=3hWQ736DLq2Oxh8pybnww2GknWtZzueB' \
    # host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id=' \
    # 'eZfr4d4wDu1Ike3Fw4KFobuF&client_secret=TEIvgHjjVRRPEtqbG9B9XtK4UzQY3wLj'
    response = requests.get(host)
    if response:
        return response.json()["access_token"]

request_url = "https://aip.baidubce.com/rest/2.0/ocr/v1/accurate_basic"
# 二进制方式打开图片文件
f = open('./images/img2txt.png', 'rb')
img = base64.b64encode(f.read())

params = {"image":img}
access_token = get_access_token()
print(access_token)
request_url = request_url + "?access_token=" + access_token
headers = {'content-type': 'application/x-www-form-urlencoded'}
response = requests.post(request_url, data=params, headers=headers)
if response:
    print(response.json())

```

5. 人机对话机器人:

```
import requests

from aip import AipImageClassify, AipSpeech

""" 你的 APPID AK SK """
APP_ID = '25211913'
API_KEY = 'H01HCGlxH564BWG6UcK8TDVv'
SECRET_KEY = 'S0gAUjjGIY4LizaBctDOGF8s4iDfOU7'

client_img = AipImageClassify(APP_ID, API_KEY, SECRET_KEY)
client_speech = AipSpeech(APP_ID, API_KEY, SECRET_KEY)

class Aip:

    def __init__(self, filePath):
        self.img = self.get_file_content(filePath)

    """ 读取图片 """

    def get_file_content(self, filePath):
        with open(filePath, 'rb') as fp:
            return fp.read()

    """ 调用通用物品和场景识别 """

    def advance_general_reg(self):
        options = {}
        options["baike_num"] = 5 # 返回百科信息的结果数
        return client_img.advancedGeneral(self.img, options)

    """ 语音合成 """

    def speech_synthesis(self):
```



```

options = {}
options['vol'] = 5 # 音量 5
options['spd'] = 3 # 语速 3
options['pit'] = 5 # 音调 5
text, file_name = self.get_information()
ret = client_speech.synthesis(text, 'zh', 1, options)
if not isinstance(ret, dict):
    with open("./audios/" + file_name + ".mp3", 'wb') as f:
        f.write(ret)

""" 返回百科信息 """

def get_information(self):
    ret = self.advance_general_reg()
    # print(ret["result"][1]["keyword"])
    # 图片描述, 返回百科描述及关键词 (作为文件命名)
    print(ret["result"][1]["baike_info"]["description"])
    return ret["result"][1]["baike_info"]["description"], ret["result"][1]["keyword"]

"""返回 access_token"""

def get_access_token(self):
    host = 'https://aip.baidubce.com/oauth/2.0/token?grant_type=client_credentials&client_id='\
'eZfr4d4wDu1lke3Fw4KFobuF&client_secret=TEIvgHjjVRRPEtqbG9B9XtK4UzQY3wLj'
    response = requests.get(host)
    if response:
        return response.json()["access_token"]

"""测试对话机器人"""

def test_access(self):
    url = 'https://aip.baidubce.com/rpc/2.0/unit/service/v3/chat?access_token=' +

```

```

self.get_access_token()

# post_data =
{"version":3.0,"service_id":"S61779","session_id":"[1127231,1127232,1127233]",
# "log_id":"7758521","request":{"terminal_id":"88888","query":"你好 a"}}
post_data =
{"version\\":\\"3.0\\",\\"service_id\\":\\"S61779\\",\\"skill_id\\":\\"[1127231,1127232,1127233]\\",\\" \
\\"session_id\\":\\"\\",\\"log_id\\":\\"7758521\\",\\"request\\":{\\"terminal_id\\":\\"88888\\",\\"query\\":\\"你好\\"}}"}

headers = {'content-type': 'application/x-www-form-urlencoded'}
response = requests.post(url, data=post_data.encode("utf-8"), headers=headers)
if response:
    print(response.json()["result"]["responses"][0]["actions"][0]["say"])
    return response.json()["result"]["responses"][0]["actions"][0]["say"]

if __name__ == '__main__':
    filePath = "./images"
    aip = Aip(filePath + "/dog.png")
    aip.get_information()
    # aip.speech_synthesis()
    # aip.test_access()

```