# 语音识别

2018年11月26日 21:43

总结：

- 只调查了stt(speech-to-text)方面，还有相对的一个方面tts(text-to-speach)
- 如果是针对这一个项目，总体来讲建议用API
  - 毕竟不是在做这方面的研究，只是用这个做我们的产品
  - 当然感兴趣想学习下请随意~
  - 好处：调用很方便，效果成熟准确率要高
  - 一些缺点：主要是在线调用非本地（不过我们设备大概总归是要联网的，根据经验延迟一般可以接受），不如自己训练高大上
- 自己训练一点儿的，能本地部署：
  - 开源项目(CMU,Kaldi)等中文可能需要自己训练，效果不清楚
  - 不过下面有个基于tensorflow写的貌似结果还迷之不错，如果真的的话倒可以考虑下

**在线API：科大讯飞，微软，百度等**

使用的感觉：

```
session_begin_params = "sub = iat, domain = iat, language = zh_cn, accent = mandarin,
sample_rate = 16000, result_type = plain, result_encoding = utf8";
sessionID = QISRSessionBegin(None,session_begin_params, byref(ret_c));
```

设置session: 调用API的账号，一些参数，文件

```
ret = QISRAudioWrite(sessionID, None,0, 4,
byref(ep_stat), byref(rec_stat));
```

返回文字结果

后面接一堆对返回结果正确性的判断，无误输出

**本地训练，基于库或者其他：**

**数据集：**

中文识别数据集

- aishell: AI SHELL公司开源178小时中文语音语料及基本训练脚本，见kaldi-master/egs/aishell
  aishell-1：http://www.openslr.org/33/
  aishell-2：需要在官网申请，禁止商用
- gale_mandarin: 中文新闻广播数据集(LDC2013S08, LDC2013S08)
- hkust: 中文电话数据集(LDC2005S15, LDC2005T32)
- **thchs30**: 清华大学30小时的数据集，http://www.openslr.org/18/

一个不知道谁做的于深度学习的识别：

https://github.com/nl8590687/ASRT_SpeechRecognition#a-deep-learning-based-chinese-speech-recognition-system

中文语音识别 80%的准确度，国内相关团队可以做到97%

**另一个基于深度学习和清华数据集的识别，似乎好一点：**
https://github.com/xxbb1234021/speech_recognition

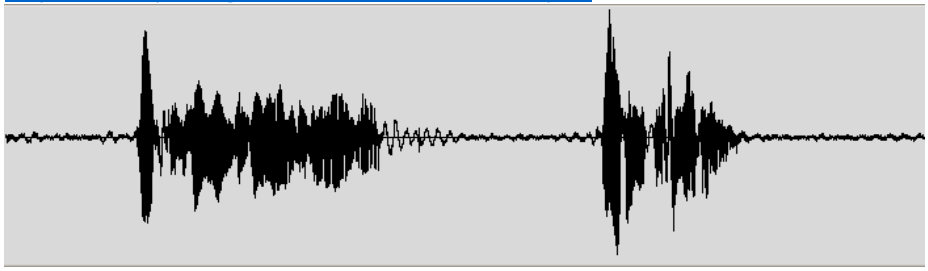
现有的语音识别开源项目：

| Toolkit | Programming languages | Development activity | Tutorials and examples | Community | Trained models |
|---|---|---|---|---|---|
| CMU Sphinx | Java, C, Python, others | +++ | +++ | +++ | English plus 10 other languages |
| Kaldi | C++, Python | +++ | ++ | +++ | Subset of English |
| HTK | C, Python | ++ | +++ | ++ | None |
| Julius | C, Python | ++ | ++ | + | Japanese |
| ISIP | C++ | ++ | ++ | + | Digits only |

后面三个大概不太适合（其中姓J的针对日文）

Kaldi据说更先进一点，每天都有更新，似乎会用到更多的深度学习方面的东西
但是对中文支持也不好，https://blog.csdn.net/shichaog/article/details/73655628有一个识别的例子但是结果不怎么样
https://github.com/kaldi-asr/kaldi

https://cmusphinx.github.io/wiki/tutorial/
**不错的教程，基本知识和使用**
基本知识：
https://cmusphinx.github.io/wiki/tutorialconcepts/



识别语音中的phone，组成subword(这里用的是syllables，Next, phones build subword units, like syllables)再组成word
中文比方识别一个字的声音，再组成词语之类的
filler和utterances的概念；根据停顿分成几个utternaces来训练
Words and other non-linguistic sounds, which we call `fillers` (breath, um, uh, cough), form `utterances`. They are separate chunks of audio between pauses. They don’ t necessary match sentences, which are more semantic concepts.

声学模型：音节的识别
音节字典：从音节到单词

语言模型：根据词的相邻出现可能性（比如跑后面可能是步，不可能是数电），选出在语境中合适的词

包括：

- Pocketsphinx — lightweight recognizer library written in C.
- Sphinxbase — support library required by Pocketsphinx
- Sphinx4 — adjustable, modifiable recognizer written in Java
- Sphinxtrain — acoustic model training tools

>
https://cmusphinx.github.io/wiki/tutorialoverview/
这里是一些资源

有关中文的资源：
CMU sphinx的官网提供了各种语言的声学模型和语言模型的下载，具体见：

http://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/
（但是没有找到）
其中也有中文的：

声学模型：zh_broadcastnews_16k_ptm256_8000.tar.bz2

语言模型：zh_broadcastnews_64000_utf8.DMP

字典文件：zh_broadcastnews_utf8.dic

（3）自行训练得到：

因为每个人的声音不一样，另外应用也不一样（所需词汇类别和不同），所以别人的模型可能在自己的语音下识别准确率不高，这样就需要训练自己的声学模型和语言模型（CMU也提供一个改进现有声学模型的方法）。因为训练需要的准备和步骤挺多的，所以这里就不写了，具体的训练方法会在下一个博文中说明。
https://blog.csdn.net/zouxy09/article/details/7942784

使用：
需要指定三个文件：
运行 pocketsphinx进行语音识别需要指定三个文件：声学模型、语言模型和字典文件。我们以第二个网上下载回来的这三个文件为例子说明如何使用他们进行语音识别。

声学模型：zh_broadcastnews_16k_ptm256_8000.tar.bz2

语言模型：zh_broadcastnews_64000_utf8.DMP

字典文件：zh_broadcastnews_utf8.dic

```java
package com.example;

import java.io.File;
import java.io.FileInputStream;
import java.io.InputStream;

import edu.cmu.sphinx.api.Configuration;
import edu.cmu.sphinx.api.SpeechResult;
import edu.cmu.sphinx.api.StreamSpeechRecognizer;

public class TranscriberDemo {

    public static void main(String[] args) throws Exception {

        Configuration configuration = new Configuration();

        configuration.setAcousticModelPath("resource:/edu/cmu/sphinx/models/en-us/en-
        configuration.setDictionaryPath("resource:/edu/cmu/sphinx/models/en-us/cmudic
        configuration.setLanguageModelPath("resource:/edu/cmu/sphinx/models/en-us/en-

        StreamSpeechRecognizer recognizer = new StreamSpeechRecognizer(configuration)
        InputStream stream = new FileInputStream(new File("test.wav"));

        recognizer.startRecognition(stream);
        SpeechResult result;
        while ((result = recognizer.getResult()) != null) {
            System.out.format("Hypothesis: %s\n", result.getHypothesis());
        }
        recognizer.stopRecognition();
    }
}
```