

中科院声学所在线识别引擎

开发测试

目录

[1.概述 5](#_Toc90310466)

[1.1简介 5](#_Toc90310467)

[1.2 读者对象 7](#_Toc90310468)

[1.3术语定义 7](#_Toc90310469)

[3.测试工程 8](#_Toc90310470)

[3.1 测试工程 8](#_Toc90310471)

[3.1.1测试工程目录结构说明 8](#_Toc90310472)

[3.1.2 测试工程编译说明 10](#_Toc90310473)

[4.测试环境 10](#_Toc90310474)

[4.1 测试工程 11](#_Toc90310475)

[4.1.1测试环境目录结构说明 11](#_Toc90310476)

[4.1.2测试环境使用说明 13](#_Toc90310477)

[4.1.3测试环境结果查看 13](#_Toc90310478)

[4.1.3测试环境配置文件 15](#_Toc90310479)

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| 1.0 | 陈向东 | 龚云波  黎塔 | 2021/04/28 |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 1.概述

1

1.1简介

“好记性不如烂笔头”，反映出人们对记录的重视程度。从传统的  
纸文化进步到计算机时代，除了手写识别，当信息量更大的时候，人们  
会选择音频和视频。虽然人们不用担心存储容量不够用，但是今后如何  
有效的利用这些多媒体的记录，成为一个突出问题。

很多人为此倾向花高价邀请速录员。一来需要成本，二来在专业领  
域，培训和保密又成为新的问题。现在，用户有了新的选择-声学所识别引擎。该识别引擎不仅能保存语音，还能把语音转化成文字与之对应保存下来。有了它，今后无论何时用户想要调出所需的记录，只要通过查找相应文字并简单点击，相关的语音片断就找到了。同时，本引擎还提供了直接识别以往录音的功能，把以前宝贵的资料统统转化成文本，给用户的媒体档案库将来查询时提供最大便利。

中科院声学所中科信利连续语音识别引擎，针对连续音频流（即来自说话人直接录入的语音，或者广播电台、电视台或其他领域的音频信号）进行识别，将音频信息自动转化成文字。覆盖汉语中绝大多数词语，适用于说普通话的任何人群。输出的结果都是汉字，兼容数字。

在输入的声音中，检测出可靠的语音，排除噪声音乐等，实时送入语音识别解码器进行识别。识别引擎把音视频中提取出的语音分成 25 毫秒一帧，提取有用特征，然后识别出一些类似拼音的结果（声学模型），再根据汉语字词句之间的搭配概率（语言模型），综合考虑。当然，考虑的越多（beam路径越大），识别准确率相对提高，同时消耗的时间就增长了；所以我们有优化策略，及时排除不可能的结果，避免系统过慢。用户可以通过调节这些参数来平衡识别质量和速度，以满足实际的需要。

最后返回的识别结果，软件以汉语中基本词语为单位给出了包含对应时间点的词序列。

需要说明的是，对识别正确率以及识别速度来说，输入语音（普通话）的质量（录音过程）是很重要的。录音时应尽可能的排除噪声和音乐、增大语音，识别效果就会好得多，因此建议用户在录音的时候选用质量较好的麦克风。

另外，系统在检测是否有语音信号时可能会带来一定的时间滞后，用户在实时录入时在每句话结束后，需要略有停顿，这样可以提高系统语音检测的速度和精度。语音检测的越好。识别引擎的识别效果也就越好。

## 1.2 读者对象

本文档是对识别引擎的内部代码流程和相关技术点的整体说明，因此读者最好满足如下几点条件后再读此文档：

* 有一定的C和C++语言基础
* 了解语音识别过程和原理
* 了解声学所识别引擎实现原理

## 1.3术语定义

TBNR，Thinkit-Broadcast-News-Recongnize：中科信利识别引擎

CN，Confuse-Network：混淆网络

VAD，Speech Activity Detection：语音活动检测

WFST, Weighted Finite-State Transducer：加权有限状态转录即

ITN,Inverse –Text-Normalization：逆文本标准化

# 3.测试工程

3

## 3.1 测试工程

测试工程组要完成对识别引擎接口的集成，实现识别引擎对录音的转写功能。提供给用户对如何集成识别引擎，引擎具体工作流程和测试一个参考实例。测试工程参考提供的ASROnlineTestEnv/Example

### 3.1.1测试工程目录结构说明

参考提供测试环境ASROnlineTestEnv/Example/TBNR\_API\_TBNRTestDemo测试工程用例。

TBNR\_API\_TBNRTestDemo目录结构说明

-API\_Include 识别引擎头文件目录

--TBNR\_API.h 识别引擎头文件

-API\_Lib 识别引擎库目录

--libTBNR\_API.so 识别引擎库

-API\_Tools 测试用例工程所需功能函数目录

-KWSAPI\_LIBLINUX 识别引擎linux环境下依赖库文件目录

-TBNR\_TKW\_Test\_demo 识别引擎测试用例源文件目录

--TBNR\_Test\_Stream

---TBNR\_Test\_Stream.cpp 识别引擎测试用例源文件

---TBNR\_Test\_Stream.makefile 识别引擎测试工程的makefile文件

---buildTest.sh 识别引擎测试工程的编译脚本

### 3.1.2 测试工程编译说明

参考提供测试环境ASROnlineTestEnv/Example/TBNR\_API\_TBNRTestDemo测试工程用例。

# 4.测试环境

4

## 4.1 测试工程

测试环境为识别引擎完成转写等功能所依赖的运行环境。识别引擎测试环境参考提供的ASROnlineTestEnv/Env

### 4.1.1测试环境目录结构说明

-bin 引擎的工作目录

--lib 识别关键词引擎库和引擎测试程序可执行文件目录

---libTBNR\_API.so 识别引擎库文件

---TBNR\_Test\_Stream 识别引擎测试程序可执行文件

---buildKWFST 建立热词模型的可执行文件(不需热词功能时不必关心该文件,引擎默认不做热词功能)

--log 识别关键词引擎日志文件目录

---DictationLog.txt 识别引擎日志文件

--run.bat 识别关键词引擎的启动脚本

--hotwords.txt 样例热词列表文件(不需热词功能时不必关心该文件,引擎默认不做热词功能,为GBK格式且为unix格式)

--tbnr\_Result.txt 识别结果文件

--exapmle.far ITN(中文数字转阿拉伯)模型

--wavlist.txt 语音任务列表

-KWSAPI\_ENV 识别引擎依赖库目录

--KWSAPI\_LINUX 识别引擎依赖库所在目录

-model 识别引擎模型字典配置文件目录

--cts\_src 识别模型存放目录

--model.eteh.dim320\_96\_64\_32\_alimid.ep25\_29.bin 识别模型

--TLG.new.bin TLG模型

--vocab.kefu\_cts.txt 识别字典

--scripts 识别引擎配置文件目录

---WFST\_onlyRec.cfg 识别模式下的识别引擎配置文件

--ElvaData 送入引擎中语音和提取特征保存目录

--hotword 热词模型生成，热词配置文件，热词依赖文件存放目录(不需热词功能时不必关心该文件,引擎默认不做热词功能)

---ConfusionMatrix.word 前端热词音素混淆文件

---hotword.cfg 前端热词配置文件

---mono.list 前端热词音素集

---syllable.word.179phones 前端热词发音文件

--punctuation 为标点模型及其配置文件

---config.cfg 标点配置文件

---model标点模型文件

--word2phone 词转发音目录

---word2phoneme\_ctstykf\_english.txt 词转发音的文件

-wav 用例语音目录

--8K\_16BIT\_PCM 8k16bitpcm格式语音目录

--8K\_8BIT\_ALAW 8k8bitalaw格式语音目录

### 4.1.2测试环境使用说明

打开引擎脚本run.sh修改相应参数：

* 修改语音任务路径参数DataDir,
* 修改引擎启动并发线程数参数linenum,
* 修改语音格式参数wavformat,改参数需要与出来的语音任务格式一致,否者识别错误;
* 可选(前端热词功能)：./lib/buildKWFST -inbfsm ../model/cts\_src/TLG.bin –key hotwords.txt --outbfsm ../model/hotword/TLG.rescore.bin

指定热词文件hotwords.txt，生成前端热词模型../model/hotword/TLG.rescore.bin，修改前端热词参数，则前端热词功能开启；

* 启动引擎启动脚本run.sh;

### 4.1.3测试环境结果查看

识别引擎的输出结果有两种格式：文本格式和分词格式,两种格式的结果内容是一样的,只是格式上的区别,文本格式结果保存引擎的工作目录下的tbnr\_Result.txt中，分词格式保存在任务语音目录下一个语音文件对应一个分词结果文件。

别引擎文本结果tbnr\_Result.txt文件格式说明:

所有任务语音的转写结果都在该文件内且一个任务语音转写结果占据一行,输出结果默认是GBK编码格式，如下所示：

你好很高心为你服务 有什么需要帮助的 (../wav/1.wav)

今天天气很好 我们打算去哪玩 (../wav/2.wav)

…

识别引擎分词结果文件格式说明:

一条语音文件对应着一个分词结果文件,如下所示：

第一行是该段语音在整通语音中的起始时间和结束时间

第二行是该段语音的转写结果,分词之间用空格隔开

第三行是转写结果中各个分词的起始时间和结束时间

第四行是该段语音转写的置信度

1.wav对应的分词结果文件为1\_sent.txt其内容：

0.1 0.7

你 好 很 高 兴 你 服 务

0.15 0.25 0.25 0.35 0.35 0.40 0.40 0.50 0.50 0.60 0.60 0.70

100.00

0.8 1.2

有 什么 需要 帮助的

0.9 1.0 1.0 1.1 1.1 1.2 1.2 1.3

100.00

…

### 4.1.3测试环境配置文件

引擎开放了大量内部参数，保证了引擎的可配性和可维护性。常见参数如下：

|  |  |  |
| --- | --- | --- |
| 参数名 | 取值 | 说明 |
| LogFileName | ./log/DictationLog.txt | 引擎日志文件 |
| LogLevel | 0 | log级别值越大输出越详细[0-3] |
| SampleRate | 8000或16000 | 声学模型训练语音的采样率。目前仅广电模型SampleRate=16000，其余模型下均为8000。 |
| isOnlineMode | true或fasle | 是否走在线语音流处理流程。true:在线；false:离线系统。 |
| NumberOfSession | >=1的整数 | 开启会话数。实际可通过TBNR\_Init()传参设定开启会话数，仅当参数<=0时, NumberOfSession才起作用。实际会话个数最多不能超过授权线数。 |
| NumberOfDecoder | >=1的整数 | 解码器个数。在线模式下，实际解码器个数=实际会话个数；离线模式下，decoderNumPerSession>0时，实际解码器个数= int (0.5 + decoderNumPerSession \* sessionNum)。实际TBNR解码器个数最大不能超过授权文件授权线数。 |
| NumberOfDecoderPerSession | >=1的整数 | 每个会话开启的解码器个数。仅对离线模式适用。 |
| CNFormat | DaLian或KWS | 引擎模式。仅做纯识别还是要做关键词检索。 |
| IndexWithRec | true或  false | 是否建索引。  CNFormat与IndexWithRec一般如下搭配使用：   * CNFormat=DaLian,IndexWithRec=false:纯识别模式 * CNFormat=KWS, IndexWithRec=false:纯关键词检索模式 * CNFormat=KWS, IndexWithRec=true:识别+检索模式 |
| isSaveData | true或  false | 是否保存分段语音。isSaveData=true,保存分段语音到参数“FeatureRootDir”所指目录下（一般位于model/ElvaData/目录下）。 |
| isSaveFeature | true或  false | 是否保存分段特征。isSaveData=true,保存分段特征到参数“FeatureRootDir”所指目录下（一般位于model/ElvaData/目录下）。 |
| KeepAllSpeech | true或  false | 是否保留无效语音(如，静音等)。  KeepAllSpeech=false,无效语音不送解码。  KeepAllSpeech=true,无效语音送解码。 |
| isDoEPD | true或  false | 是否做分段（端点检测）。 |
| VadFormat | oldEPD或BC | 端点检测算法。BC:广电引擎端点检测；oldEPD:EPD端点检测。 |
| isHighPass | true或  false | 是否做高通滤波 |
| DetectRingNum | >=0的整数 | 彩铃检测开关。DetectRingNum=0时，不做彩铃检测。DetectRingNum>0时，做彩铃检测。 |
| PrintSegTimeOFWord | true或  false | PrintSegTimeOFWord =true: 转写候选结果中词的时间是相对于整条语音的绝对时间；  PrintSegTimeOFWord =false：词的时间是相对于句的时间。 |
| PrintPunctuation | true或false | 是否输出标点。如果PrintPunctuation=true，则必须设置PrintSegTimeOFWord=true。 |
| PrintPinyin | true或false | 识别结果带拼音输出。如果PrintPinyin=true，必须设置字典WordPhonemePinyinDict。且只有当解码器版本支持拼音输出时，最终才能够输出拼音。 |
| isGBKRecResConverUTF8 | true或false | 识别结果进行GBK转U8 |
| isHoldWavDataOnline | true或false | 在线hold数据延缓解码 |
| HoldWavDataBySampleNumOnline |  | Hold数据大小（采样数） |
| DictFileName |  | 识别用字典，含词语及其对应音素发言。该字典跟模型一一对应，更换模型时，注意更换相应字典。 |
| isAddPhonemeEnd2End | true/false | 是否输出拼音 |
| WordPhonemePinyinDict |  | 识别结果输出拼音时所用字典，含词语、对应音素及拼音。 |
| HotWordOn | true/false | 后端热词开关 |
| HotWordConfigFile |  | 后端热词配置文件。 |
| isReturnMidRecRes | true/false | 是否实时返回识别结果 |
| isTestEPD | true/false | 仅仅测试vad |
| ModelFile |  | 端到端模型文件 |
| VocabularyFile | ../ElvaData | 字典文件。 |
| wfst |  | TLG模型文件 |
| key\_wfst |  | 前端热词模型文件 |
| keywt | 0.00~5.00 | 前端热词权重 |
| harmonicMinNumber\_EPD |  | 最小谐波大小(帧) |
| MAX\_SILENCE\_EPD= | 20 | 最大静音长度(帧) |
| ENERGY\_DECREASE\_END |  | 最小谐波长度(帧) |
| ENERGY\_DECREASE\_START | 1 | 检测起点能量 |
| ENERGY\_DECREASE\_END | 1 | 检测尾点能量 |