

中科院声学所叠音检测引擎

代码模块接口设计说明

目录

[1.概述 5](#_Toc89955995)

[1.1简介 5](#_Toc89955996)

[1.2 读者对象 5](#_Toc89955997)

[1.3术语定义 6](#_Toc89955998)

[2.总体设计 6](#_Toc89955999)

[3.接口设计 10](#_Toc89956000)

[3.1 引擎接口设计 10](#_Toc89956001)

[3.1.1 叠音检测引擎调用流程 10](#_Toc89956002)

[3.1.2初始化引擎 10](#_Toc89956003)

[3.1.3 叠音检测 11](#_Toc89956004)

[3.1.4 结果返回 12](#_Toc89956005)

版 本 历 史

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 版本/状态 | 作者 | 参与者 | 起止日期 | 备注 |
| 1.0 | 张英楠 |  | 2021/04/28 |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 1.概述

1

1.1简介

叠音自动检测在语音技术领域中具有十分重要的地位。该技术是指从声音信号中寻找出多个说话人语音的重叠部分。进行有效的叠音自动检测能够对声音信号中多个说话人互相打断对话的情况进行检测。

## 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

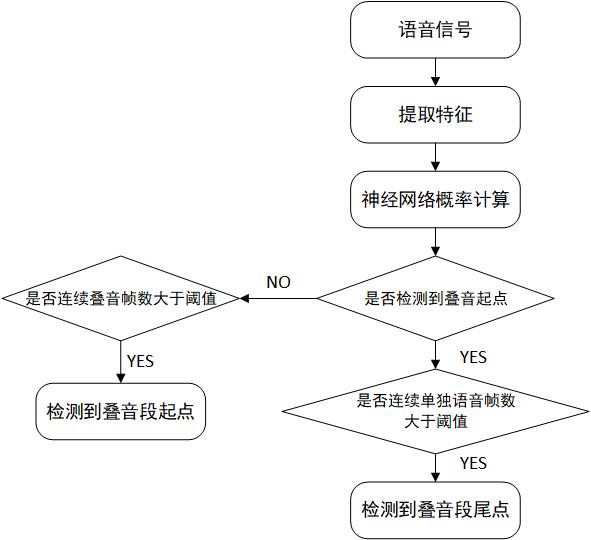
## 1.3术语定义

DNN(Deep Neural Networks）：深度神经网络

# 2.总体设计

2

叠音检测引擎是基于深度神经网络建模的方法对录入连续的语音数据进行有效的叠音自动检测并告知叠音片段语音数据的起始位置和结束位置，从而可以利用检测出有效的叠音数据为某些特定场景应用提供信息，总体流程图如下：



语音数据送入引擎后，经过“特征提取->DNN概率计算 ->结果判决”，最终得到叠音的起尾时间点。

* 特征提取：输入的语音数据分别经过预加重、分帧、加窗、FFT、梅尔滤波、对数运算、DCT后形成每帧40维的MFCC语音特征，语音特征数据将被送入DNN概率计算模块；
* DNN概率计算：在引擎初始化阶段加载DNN模型，构建神经网络；将特征送入神经网络中，计算出后验概率，送入结果判决模块；
* 结果判决：根据连续的语音帧或者叠音帧是否超过阈值，来判断叠音的开始以及结束。

# 3.接口设计

3

## 3.1 引擎接口设计

## 3.1.1 叠音检测引擎调用流程

引擎的流程如下:

1. 初始化引擎
2. 读入语音数据，并进行DNN判别，以链表的形式返回叠音的起尾点；

## 3.1.2初始化引擎

|  |  |
| --- | --- |
| 函数原型 | **bool InitDetectOverlap(const char \*systemDir, const char \*cfg);** |
| 功能 | 读取配置并初始化引擎 |
| 参数说明 | const char \*systemDir：系统目录地址 |
| const char \*cfg: 配置文件名 |
| 返回值 | 成功返回true，失败返回false |
| 说明 |  |

## 3.1.3 叠音检测

|  |  |
| --- | --- |
| 函数原型 | **bool Process(float \*newSet, int nFrameNum, int startFrm, int nDuration, short nFrameSize);** |
| 功能 | 接收数据并进行DNN判别 |
| 参数说明 | **float \*newSet：语音特征** |
| **int nFrameNum：语音特征帧数** |
| **int startFrm：输入语音起点绝对时间（在整通语音中）** |
| **int nDuration, short nFrameSize：特征系数** |
|  |
| 返回值 | 成功返回true，失败返回false |
| 说明 |  |

## 3.1.4 结果返回

|  |  |
| --- | --- |
| 函数原型 | **textResult \*GetResult();** |
| 功能 | 返回结果链表 |
| 参数说明 |  |
| 返回值 | **textResult链表头** |
| 说明 | typedef struct textResult\_{  char pText[128]; //存储输出结果  struct textResult\_ \*next;  }textResult; |