Project 1：基于统计模型分类器和语音短时特征的语音端点检测

518021910698 薛春宇

Abstract

语音端点检测 *(Voice Activity Detection)* 的目的是对连续语音信号中语音和非语音的区域进行区分，其准确性直接影响到整个语音识别系统的性能。本项目中基于统计模型分类器的 *VAD* 算法将语音检测视作二分类问题，通过语音的短时时域/频域特征构造特征向量，使用高斯混合模型 (*GMM*) 构建分类器，并基于 *train* 训练集进行训练和指标评估，基于 *dev* 开发集进行 *AUC*、*EER* 指标验证，采用有限状态机架构以达到端点检测的目的。综合其在 *test* 测试集上的表现得出，该模型为语音端点检测提供了一个简单高效的方案。

Keyword: 语音端点检测，语音短时特征，高斯混合模型，指标评估，有限状态机

1. Introduction

语音端点检测 (VAD) 通过对连续语音信号中的语音/非语音区域进行探测和标注，为后续语音识别的准确性提供了坚实的基础。准确可靠的 VAD 可以减少语音识别系统的计算量和能耗需求，进而改善上层系统的性能。大多数 VAD 系统通过监控一个指标，并将其与设定好的阈值作比较，以此观察该信号是否为语音信号 [1]，这个阈值可以通过启发式算法或自适应算法选择 [2]。这个指标最常见的是语音的短时能量，可以是梅尔频率倒谱系数 MFCC[3]，也可以是几种语音信号短时特征的组合。

本文首先在第 2 节中介绍一些用于构建模型特征向量的语音短时时域和频域特征，其中，时域特征包括短时能量 (Short Time Energy) 和短时过零率 (Zero-crossing Rate)，频域特征包括梅尔频率倒谱系数 (Mel-Frequency Cepstral Coefficients) 和感知线性预测系数 (Perceptual Linear Predictive)，上述特征均具有 “短时平稳” 的特性。基于上述特征，本文将在第 3 节中使用机器学习的方法构建了一个高斯混合模型统计分类器，并在 train 训练集上进行模型训练，以及包含 AUC 和 EER 指标在内的模型评估。在上述统计分类模型之上，本文会在第 4 节通过简单有限状态机的方式，实现针对不同语音信号进行端点检测的目标。第 5 节中，在 dev 开发集和 test 测试集上的实验结果表明，该 VAD 模型在不同声音、不同背景噪声之下，均具有较好的端点检测能力。

2. 语音信号短时特征

语音信号属于短时平稳信号，加窗分帧后具有短时平稳的性质，这为我们构造语音特征向量，训练统计模型分类器提供了理论支持。语音信号处理中一般采用 10 - 30ms 的帧长，在本文中，我们选择 30ms 作为单个语音帧的长度。同时，为充分利用语音信号的相关性，我们取 15ms 作为帧移，使得相邻帧之间具有一定的交叠。

对于语音短时特征的分析又可以分为时域和频域两个方向。短时时域特征主要体现语音信号在能量、清浊上的特征，典型代表是短时能量 STE 和短时过零率 ZCR，前者是语音信号一帧内采样点的平方和，后者是一帧内信号正负反复的次数。

短时频域特征则主要体现语音信号频率、能量分布方面的特征，典型代表是梅尔频率倒谱系数 MFCC，滤波器组 FBank，线性预测系数 LPC 和在其上的感知线性预测系数 PLP 等。

MFCC 是在 Mel 标度频率域提取出的倒谱参数，是一种在自动语音和说话人识别中得到广泛应用的特征。Mel 标度描述了人耳频率的非线性特性，它与频率的关系可用下式近似表示：

$$Mel(f) = 2595 \times lg(1 + \frac{f}{700}) \tag{1}$$

上式中 f 为频率，单位是 Hz。在 Mel 频率域内，人对音调的感知度为线性关系。滤波器组 FBank[1] 与 MFCC 的计算过程除 IDFT 外基本一

致，且包含信息较多，计算量较小，但其相邻滤波器组存在重叠，特征相关性较高，在语音识别中的判别度要弱于 MFCC，因此本文选择使用前 13 维 MFCC 而非 FBank 来构成部分特征向量。

短时频域特征的另一大类是线性预测系数 LPC 及结合人耳感知特性得到的感知线性预测系数 PLP。由于语音信号的发音特性，提取特征后帧与帧之间是不独立的，因此一个语音采样能够用过去若干个语音采样的线性组合来逼近。在线性预测分析中，我们可以使用全极点滤波器为声道响应函数建模，通过线性预测得到的采样在最小均方误差意义上能够逼近实际语音采样，该线性组合中所用的加权系数即为 LPC。

PLP 是一种基于听觉模型的特征参数，将人耳听觉试验获得的结论，通过近似计算的方法进行工程化处理，应用到频谱分析中，将语音信号经听觉模型处理后得到的信号代理传统 LPC 分析所使用的时域信号。PLP 主要由频谱分析、临界频带分析、等响度预加重和强度-响度转换组成 [4]。由于考虑了人耳的听觉特性，PLP 在抗噪语音特征提取中具有良好的性能，因此本文选取语音信号的前 15 维 PLP 系数构成部分特征向量。

3. 基于高斯混合的统计模型分类器

本节中，我们将基于 train 训练集介绍包含数据预处理、特征提取、高斯混合模型分类器算法、指标评估在内的完整流程。

3.1. 数据预处理及特征提取

本文使用 VAD 数据集对分类器的进行训练和测试，包括 train 训练集、dev 开发集和 test 测试集三个子数据集。其中，train 训练集（3600 个 wavs）将用作 GMM 模型的训练和初步评估，dev 开发集（500 个 wavs）用作模型泛化性的验证，test 测试集（1000 个 wavs）用作最后的结果预测。注意到，本文中的 GMM 模型采用的特征向量维度为 30，其中包括 15 个 PLP 系数，13 个 MFCC 系数，1 个 STE 参数和 1 个 ZCR 参数。

[1]FBank: https://www.jianshu.com/p/b25abb28b6f8

每个子数据集包括 wav 格式的数据文件和对应的 txt 标签两个部分，在读入前会针对文件名分别进行字典排序，从而保证数据和标签对应关系的正确性。对于原始 wav 文件，我们首先将语音信号进行归一化、加窗分帧等基本操作，随后逐帧进行短时时域 (STE, ZCR) 和频域 (MFCC, PLP) 特征的提取，并以此为基础构造特征向量，与 wav 文件内的帧一一对应。在分帧操作中，我们选择 30ms 的帧长和 15ms 的帧移。对于 label 文件，我们首先逐行读取 txt 文件，进行起始时间点的格式划分，并将语音端点根据已有的帧长、帧移从以时间为单位对应到以帧为单位，并将位于起止端点内的帧全部打上 1 的标签，代表这些帧内均存在语音事件。

对于已获得的 30 维特征向量，我们需要使用如下公式来进行均值归一化：

$$f_{ij} = \frac{f_{ij} - \bar{f}_j}{\Delta f_j} \tag{2}$$

其中，f_{ij} 表示第 i 个帧中的第 j 项特征，其中 $1 \le j \le 30$。$\bar{f}_j$ 表示第 j 项特征在全部帧上的均值，Δf_j 表示第 j 项特征在全部帧上最大值和最小值的差。

由于整个数据集单次特征提取仍会花费较长时间，我们选择将上述方法生成的特征向量，label，以及模型统计分布（$\bar{f}_j$ 和 Δf_j）分别保存为单独的 npy 文件，以便之后在模型训练、评估和预测中直接调用。

3.2. 高斯混合模型分类器算法

高斯混合模型 (Gaussian Mixed Model) 简称 GMM，是一种业界广泛使用的聚类算法，指由多个高斯分布函数线性组合形成的统计模型，理论上可以拟合出任意类型的分布，通常用于解决同一集合下数据的多分布情况。设有随机变量 $\boldsymbol{X}$，则混合高斯模型可以如下表示：

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k \boldsymbol{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma_k}) \tag{3}$$

其中，$\boldsymbol{N}(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma_k})$ 称为混合模型中的第 k

个分量，K 为分量数，π_k 是混合系数，且满足：$\sum_{k=1}^{K}\pi_k=1$，$1\leq\pi_k\leq 1$，可以认为 π_k 即为第 k 个分量的权重系数。

为了对 GMM 模型进行训练，我们需要使用期望最大化算法 (Expectation-Maximization Algorithm) 来对估计 GMM 的参数[2]。EM 算法分成两步，分别是求出待估计参数的粗略值，以及使用该粗略值最大化似然函数。

GMM 模型中有三个参数需要估计，分别是 $\boldsymbol{\pi}$、$\boldsymbol{\mu}$ 和 $\boldsymbol{\Sigma}$，将 (3) 式改写为如下形式：

$$p(\boldsymbol{x}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})=\sum_{k=1}^{K}\pi_k\boldsymbol{N}(\boldsymbol{x}|\boldsymbol{\mu}_k,\boldsymbol{\Sigma_k}) \tag{4}$$

首先，在 E step 中，我们根据当前的 $\boldsymbol{\pi}_k$、$\boldsymbol{\mu}_k$ 和 $\boldsymbol{\Sigma}_k$ 计算后验概率 $\gamma(z_{nk})$：

$$\gamma(z_{nk})=\frac{\pi_k\boldsymbol{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n,\boldsymbol{\Sigma_n})}{\sum_{j=1}^{K}\pi_j\boldsymbol{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_j,\boldsymbol{\Sigma_j})} \tag{5}$$

其次，在 M step 中，我们根据 (5) 式中的 $\gamma(z_{nk})$ 计算新的 $\boldsymbol{\pi}_k$、$\boldsymbol{\mu}_k$ 和 $\boldsymbol{\Sigma}_k$：

$$\boldsymbol{\mu}_k^{new}=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\boldsymbol{x}_n \tag{6}$$

$$\boldsymbol{\Sigma}_k^{new}=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})(\boldsymbol{x}_n-\boldsymbol{\mu}_k^{new})(\boldsymbol{x}_n-\boldsymbol{\mu}_k^{new})^T \tag{7}$$

$$\boldsymbol{\pi}_k^{new}=\frac{N_k}{N} \tag{8}$$

其中，$N_k=\sum_{n=1}^{N}\gamma(z_{nk})$。最后，我们需要计算 (4) 的对数似然函数：

$$\ln p(\boldsymbol{x}|\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})=\sum_{n=1}^{N}\ln\{\sum_{k=1}^{K}\pi_k\boldsymbol{N}(\boldsymbol{x}_k|\boldsymbol{\mu}_k,\boldsymbol{\Sigma_k})\} \tag{9}$$

通过检查参数是否收敛或对数似然函数是否收敛，我们可以判断 GMM 模型的训练是否完成。

本文通过如下方式构造 VAD 模型：

- 配置 GMM 模型的参数，分量数 $K=16$，参数初始化方式选择 k-means，最大迭代次数为 500，并创建两个相同配置的 GMM，分别命名为 pos_GMM 和 neg_GMM；
- 将处理好的训练集样本逐帧划分为正、负样本两类，分别投入两个 GMM 进行模型训练；
- 预测输出时，将一条处理完成的 wav 逐帧投入两个 GMM 中，分别求出该帧信号的正、负似然度；逐帧比较两个似然度的相对大小，若正样本 GMM 对该帧的似然度较大，则将其预测为正样本，反之为负样本；
- 针对得到的预测结果（二值序列）进行 kernel size 为 17 的中值滤波。

注意到，上述 VAD 模型中的 GMM 调用了 sklearn 库进行实现，因此可以直接调用 model.fit() 成员函数来进行模型的训练。

为了进行模型训练，我们首先从保存路径中读取经过预处理得到的特征向量，并将所有 wav 文件的特征向量混合进行 shuffle，组合成 m 个维度为 $n\times 1$ 的特征向量一维列表，每个列表为一个 batch。对 label 也进行相同的操作，注意帧和 label 仍保持严格的对应关系。随着 m 取值的不同，模型的训练速度和性能也会有显著差异，这部分内容将在 3.3 节中的 big batch GMM ($m=1$) 与 mini batch GMM ($m>1$) 的对比中详细说明，以下默认为 big batch。本模块的核心代码结构如下：

```
# 创建正、负样本两个GMM
pos_GMM, neg_GMM = GaussianMixture(K=16,
    init_params="kmeans", max_iter=500)
# 读取预处理后的特征向量和标签
train_X = read([features_path])
train_Y = read([labels_path])
# 划分为正负样本集
pos_features, neg_features = classify_batch(
    features_vector)
# 模型训练
pos_GMM.fit(pos_features)
neg_GMM.fit(neg_features)
# 计算似然度，预测
pred = []
for i in range(len(train_X)):
    pos_likelihood = pos_GMM.score_samples(train_X
    [i])
    neg_likelihood = neg_GMM.score_samples(train_X
    [i])
```

[2]EM 算法：https://blog.csdn.net/jinping_shi/article/details/59613054/

```
    if pos_likelihood < neg_likelihood:
        # Silence state
        pred.append(0)
    else:
        # Speech state
        pred.append(1)
# 对预测序列进行中值滤波
pred = medfilt(pred, kernel_size)
```

3.3. 指标评估

本节中我们将介绍在上述模型训练完成后，利用 ROC 曲线下的面积 AUC 和等错误率 EER 指标来进行模型正确性评估的方法。ROC 曲线即接受者操作特征曲线，指在特定刺激条件下，以受试者在不同判断标准下所得的误报率 FPR 为横坐标，召回率 TPR 为纵坐标得到的曲线，值越大说明模型越好。EER 等于 ROC 上 FPR 和 TPR 相同时的取值，值越小说明模型越好。

训练集指标评估的具体实现方法是，遍历预处理得到的特征向量集及相应的标签集，作为整体计算 AUC 和 EER 并作为评估的统计指标。具体的计算过程被封装在 get_metrics() 函数中，通过 skearn.metrics 方法下的 roc_curve 和 auc 函数实现。

经过指标评估，VAD 模型运行结果如图1所示，训练共花费约 1.77h，$AUC = 0.9502$，$EER = 0.0830$。

```
Iteration: 2200 / 3600 rounds | Current average AUC: 0.9478666657356005 | Current average EER: 0.08797822431644565
Iteration: 2300 / 3600 rounds | Current average AUC: 0.9483156046840842 | Current average EER: 0.08707692914181527
Iteration: 2400 / 3600 rounds | Current average AUC: 0.9482782736229588 | Current average EER: 0.08727851554882694
Iteration: 2500 / 3600 rounds | Current average AUC: 0.9486473332754604 | Current average EER: 0.08658126210545999
Iteration: 2600 / 3600 rounds | Current average AUC: 0.9490565449419061 | Current average EER: 0.08568669922249439
Iteration: 2700 / 3600 rounds | Current average AUC: 0.9489275909097709 | Current average EER: 0.08580282715308335
Iteration: 2800 / 3600 rounds | Current average AUC: 0.9490865450192457 | Current average EER: 0.08542505890468384
Iteration: 2900 / 3600 rounds | Current average AUC: 0.9494212088553999 | Current average EER: 0.08478411280862648
Iteration: 3000 / 3600 rounds | Current average AUC: 0.949677978881047 | Current average EER: 0.08432658415673167
Iteration: 3100 / 3600 rounds | Current average AUC: 0.9494171148506901 | Current average EER: 0.08481413029820944
Iteration: 3200 / 3600 rounds | Current average AUC: 0.9496572492469747 | Current average EER: 0.08428940589501051
Iteration: 3300 / 3600 rounds | Current average AUC: 0.9499544250269539 | Current average EER: 0.08363921743223507
Iteration: 3400 / 3600 rounds | Current average AUC: 0.9501286058904033 | Current average EER: 0.08321303836911804
Iteration: 3500 / 3600 rounds | Current average AUC: 0.9502461624432226 | Current average EER: 0.08298304328095896
------------------------------------------------------------
Saving model, means and delta_std...
GMM train completed! | Total time: 6356.49 seconds | Train AUC: 0.9502 | Train EER: 0.0830

Process finished with exit code 0
```

图 1: *train* 训练集指标评估结果

单独取 train 训练集字典排序后的第一个 wav 文件，绘制 ROC 曲线如图2：

为了进一步探索模型后处理（中值平滑）对 AUC 和 EER 指标的影响，我们针对正、负模型似然度和预测结果是否进行中值平滑设计如下对比实验，其中 G1 表示仅对预测结果进行中值滤波，G2 表示仅对正、负似然度进行中值滤波，G3 表示二者同时进行，G4 表示不经过处理。注意，为了加快实验效率，我们设置 GMM 的最大迭代次数为 20，对比结果见表 1。

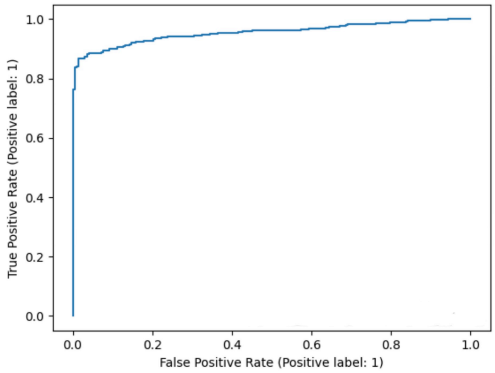

图 2: *train* 训练集指标评估结果

表 1: 中值滤波对指标评估的影响对比实验

	G1	G2	G3	G4
AUC	0.9467	0.9262	0.9349	0.9312
EER	0.1234	0.1565	0.1293	0.1381

基于上述结果得出结论：对 VAD 模型预测结果进行中值平滑有利于提高预测准确率，而对正、负模型输出的似然度进行中值平滑会造成 AUC 和 EER 指标的降低。分析原因可知，中值平滑能够消除预测结果中高频变化的噪声，且语音信号具有连续性质，因此能够提高预测准确率；然而，中值平滑也会破坏正、负模型输出似然度的相对大小，造成预测信息的丢失，因此会使得模型的性能下降。

此外，我们测试 big batch 和 mini batch 对模型性能的影响。big batch 的模型结果见图 1，mini batch 的模型结果见图 3。其中，我们设置 batch size 为 1024，GMM 模型的 warm_start 为 True，划分整个 train 训练集进行批处理训练。

经指标评估，mini batch 下的 VAD 模型训练共花费约 0.69h，$AUC = 0.8528$，$EER = 0.2721$。显然，mini batch 的训练速度要显著快于 big batch，但缺点是模型性能较差，AUC 等评估指标远差于 big batch。因此，在以最大化模型性能为目标的前提下，一般选择 big batch 来进行 GMM

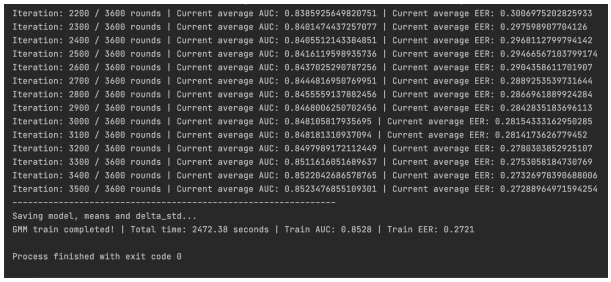

图 3: *Mini batch* 指标评估结果

的拟合。

注意，上述指标评估和模型训练均基于 train 训练集，在第 5 节中会基于 dev 开发集进行指标验证，以验证模型在不同数据集上的泛化性。

4. 基于 GMM 统计分类器的有限状态机

有限状态机是一种用来进行对象行为建模的工具，其作用主要是描述对象在其生命周期内所经历的状态序列，以及如何响应来自外界的各种事件。本节将介绍基于上述理论实现的 GMM 统计分类器，对语音端点检测中的事件进行抽象化，构建有效的有限状态机 FSM，如图 4 所示。

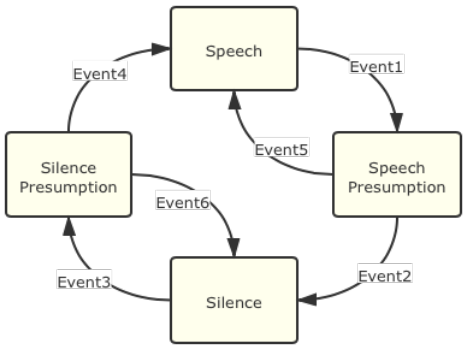

图 4: *FSM* 的状态转移关系

为了详细描述 VAD 模型预测的状态转移情况，FSM 包含 4 种状态，分别是 Silence、Silence Presumption、Speech 和 Speech Presumption。FSM 的输入为正、负样本模型输出的似然度，并进行逐帧检测。若某帧的正样本似然度大于负样本似然度，则进入 Speech Presumption 状态 (Event1)；否则进入 Silence Presumption 状态 (Event3)。对该帧及其邻域进行中值滤波，若本处于 Speech Presumption 状态，滤波后 pred[i] 变为 0，则触发 Event2 进入 Silence 状态，否则触发 Event5 进入 Speech 状态；若本处于 Silence Presumption 状态，滤波后 pred[i] 变为 1，则触发 Event4 进入 Speech 状态，否则触发 Event6 进入 Silence 状态。

最终，若该帧在 FSM 中的状态停留在 Speech，则输出 1，预测为 speech state；若停留在 Silence，则输出 0，预测为 silence state。

5. 实验和结果

本节分为两个部分，分别是基于 dev 开发集的模型指标验证，以及基于 test 测试集的语音端点预测。

5.1. 基于 dev 开发集的模型指标验证

由于 3.3 节中的指标评估仍是基于 train 训练集实现的，并不能很好地体现模型的泛化性，因此本模块将对 dev 开发集中的样本进行预处理、特征提取和结果预测，并对预测结果计算平均 AUC 和 EER 指标，得到 $AUC = 0.9459$，$ERR = 0.0906$，结果见图5：

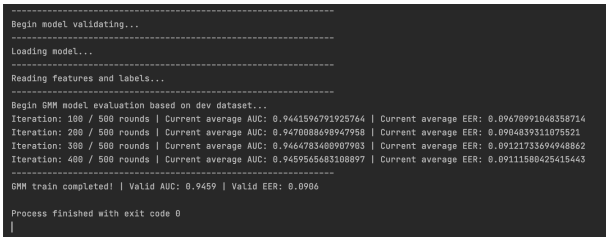

图 5: *dev* 开发集指标验证

注意到，该结果与直接在训练集上评估 (AUC 0.9502，EER 0.0830) 稍有降低，但在可接受范围内，仍具有很好的模型正确性。

5.2. 基于 test 测试集的语音端点预测

本模块对共包含 1000 个 wav 样本的 test 测试集分别进行预处理、特征提取和结果预测，并将预测的结果从以帧为单位转化到以秒为单位，

再依据数据集 label 文件的格式写入到 “./output/test_prediction.txt” 目录下，以便后续的测试和评估。通过随机选择一个 wav 文件，绘制相应的预测结果（如图 6 和 7），并与对应的 wav 文件进行人工比较可以大致验证本模型预测的正确性。

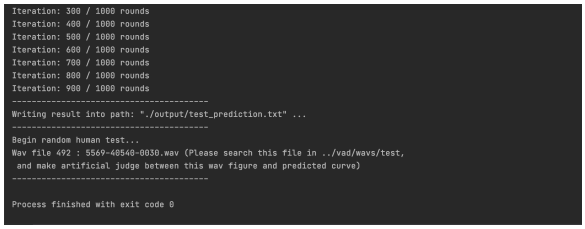

图 6: 人工检测测试集预测的准确性 *(1)*

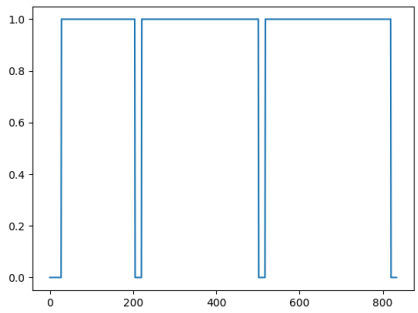

图 7: 人工检测测试集预测的准确性 *(2)*

6. 总结

本文提出了一种语音端点检测的可行性方案，通过基于语音时域和频域的短时特征构造特征向量，使用高斯混合模型构建统计分类器，并利用有限状态机 FSM 实现语音信号的端点检测。实验结果表明，该模型在拟合收敛的条件下能够达到很高的预测准确度。

7. 参考文献

1. L.R. Rabiner and R.W. Schafer. Digital processing of speech signals. Prentice-Hall, London, 1978.

2. S.Gokhun Tanyer and Hamza Ozer, Voice Activity Detection in Nonstationary Noise, IEEE Trans. Sp. Au. Proc, vol. 8, no. 4, pp 479-482, Jul. 2000

3. J. Ling, S. Sun, J. Zhu and X. Liu, ”Speaker Recognition with VAD,” 2009 Second Pacific-Asia Conference on Web Mining and Web-based Application, 2009, pp. 313-315, doi: 10.1109/WMWA.2009.59.

4. 魏艳，张雪英. 噪声条件下的语音特征 PLP 参数的提取 [J]. 太原理工大学学报, 2009(03):222-224.