中英文孤立词的识别

金熙森
复旦大学，计算机科学与技术学院
15307130053@fudan.edu.cn

摘要—本次数字信号处理课程项目中，我使用多种语音信号处理的技术以及神经网络分类器构建孤立词识别模型。我使用自适应层次 RNN 模型作为分类器骨架，并使用基于基音频率的 SVM 模型进一步提升混淆词的区分准确率，并对其中的技术进行了深入的探讨。同时，我对去除说话人不同对分类带来的影响方面做了一定的探索。实验表明，我们的模型能在验证集上获得 97.1% 的准确率。

Index Terms—孤立词识别，循环神经网络，基音频率提取，对抗网络任务

1 简介

给定中英文孤立词的音频，中英文孤立词识别任务旨在正确地分类语音包含的单词。孤立词的识别任务历史悠久，可以追溯到几十年前。研究者们提出了模板匹配模型，统计分类器的模型，以及基于深度学习的模型。尽管孤立词的任务是语音识别中最基础的任务，但是这并不代表以上技术可以在准确率上达到完美——即便深度学习模型目前获得了空前的发展，某些模型甚至可以不经过频谱变换直接从时间域的波形提取特征，他们也不能达到 100% 的准确率。噪音环境，人与人的发声差异，以及中文中大量存在的仅音调不同的单词的识别仍然是具有挑战性的待解决的问题。在报告中，我们可以看到就算最新的深度学习技术，也不能总是完美地区分“北京”与“背景”这两个单词。这告诉我们，为了处理孤立词识别的问题，我们不但需要统计学习相关的知识，还要对信号处理，语音的形成有比较透彻的理解。

本次研究基于一个给定的数据集开展实验。这一数据集包含了长度约为 2 秒的来自不同说话人的单词朗读声音，共 30 人参与了录制，每个词读 20 遍，共 20 个不同的单词。这些单词分别为：语音，余音，识别，失败，中国，忠告，北京，背景，上海，商行，复旦，饭店，Speech，Speaker，Signal，File，Print，Open，Close，Project。可以看出，这个数据集中也包含了一些容易混淆的单词对。我们重点考虑以下的研究问题：

- RQ1: 孤立词模型识别选用怎样的模型与特征进行分类较为合适？
- RQ2：如何解决仅音调不同的单词的识别问题？
- RQ3: 如何解决训练/测试说话人不同的问题？

本次工作中，识别模型的核心是基于循环神经网络的模型，并以基于特征工程的模型为辅。这篇文章包括了项目过程中所有对于模型改进的尝试。总而言之，本实验报告相近的记录的我对这一问题进行的探索。实验的结果十分具有启发性且值得思考。

2 预处理与特征提取

2.1 端点检测

端点检测是语音检测中的第一个环节。由于端点一旦确定后再后续的算法中几乎不会发生改变，因此端点检测是整个语音识别系统中至关重要的一环。端点检测的方法有许多种，大多需要设定一个阈值并进行一系列的条件判定，条件与阈值的确定没有一个统一的标准，确定起来灵活但是调参较为困难。本次实验中提出两种模型。其中第一个模型依靠短时平均幅值与短时平均过零率进行判别，记为 AZ(Amplitude&Zero cross rate) 模型；第二个模型依靠短时平均幅值，短时平均过零率与短时自相关进行判别，记为 AA 模型（Amplitude&Auto-correlation）模型。每一个模型的参数都经过了细致的参数选择。

AZ 模型的算法流程如下。

1) 将语音分帧，并求出每帧的平均幅值与平均过零率。帧长为 20ms, 帧移位 10ms，使用矩形窗截取。
2) 设定幅值高阈值 M_H。M_H 设置为语音段最大幅值的 0.25 倍。
3) 基于无声段的幅值设定幅值的低阈值。为了设定低阈值，我们假定语音的前 0.1 秒是无声的。然而，实验中发现许多录制的声音前 0.1 秒有明显的鼠标点击声音，因此，后 0.1 秒的语音也被考虑为无声段。为了减轻鼠标点击声音等“突发噪音”的影响，我们舍弃幅值大于 90%-分位数的点。之后，对这些无声段的幅值求平均值与标准差，记为 μ_0,σ_0. M_L 设定为 $\mu_0 + 3\sigma_0$。这里，我们假设无声段的声音的幅值满足正态分布。

4) 由前向后扫描每个帧，直到找到第一个幅值大于 M_H 的点。分别向左与向右延伸，直到幅值小于等于 M_L。至此，通过幅值进行端点判定的过程结束。
5) 检测合理性。若语音段小于 0.5 秒，缩小 M_H 为最大幅值的 0.125 倍，重新运行以上过程。缩小 M_H 的操作最多做一次。
6) 设定过零率阈值。通过与上述过程相同的方法估计无声段，并计算无声段的平均过零率。计算这些帧过零率的平均值与方差 μ_1 与 σ_1. 将 Z_L 设定为 $\mu_1+3\sigma_1$。最多向前向后搜索 25ms.
7) 从基于幅值获得的端点向两端扫描，直到过零率小于 Z_L. 输出两个端点。
8) 由于一段语音中的一个词可能包含多个字，因此可能得到多个语音段。这种情况下，通过选取最小的开始端点与最大的结束端点将所得到的语音段合并。

以上算法已经可以得到非常好的效果。然而，它不能很好地处理以下几种情形。第一，语音段开始十分安静。由于我们使用 $\mu_0+3\sigma_0$ 作为低阈值进行端点检测，当 μ_0 非常小时，得到的语音段包含额外的片段。第二，语音段后紧跟一些噪声。常见的有呼气的声音以及其他麦克风的杂音。这些情形少量存在于整个数据集中。这些情况有一个共同的特点：幅值够大的片段实际上并不是语音段。针对这些片段，使用自相关函数进行判别将会是一个很好的办法，因为呼气声，背景噪音可以近似认为是白噪声，自相关函数远远小于正常的语音段。此外，在之后的算法中可能仅提取浊音段进行分析（例如基音频率检测）。为此，本人设计了基于幅值和自相关函数的 AA 模型。AA 模型的算法流程如下。

1) 将语音分帧，并求出每帧的平均幅值与平均过零率。帧长为 20ms, 帧移位 10ms，使用矩形窗截取。
2) 记帧内自相关函数之和为 $R(n)$，采样率为 $rate$。对每一个帧，计算短时能量，即 $R(0)$。计算偏移量为 $rate/500$ 至 $rate/50$ 的自相关函数之和，即 $R(n), n \in [rate/500, rate/50]$。事实上 50Hz，500Hz 代表了常见的基音频率。这个阈值能够帮助我们很好地判定浊音段。
3) 设定幅值高阈值 M_H。M_H 设置为语音段最大幅值的 0.125 倍（而不是 0.25 倍）。M_H 的设置较为宽松，因为自相关函数可以帮助算法过滤幅值大的噪声片段。
4) 基于无声段的幅值设定幅值的低阈值 M_L。方法与 AZ 模型相同。
5) 由前向后扫描每个帧，直到找到第一个幅值大于 M_H 的点。分别向左与向右延伸，直到幅值小于等于 M_L，或者 $maxR(n)|n \in [rate/500, rate/50] < 0.55R(0)$。
6) 由于一段语音中的一个词可能包含多个字，因此可能得到多个语音段。这种情况下，通过选取最小的开始端点与最大的结束端点将所得到的语音段合并。

AA 模型与 AZ 模型的设计目标是不同的。AZ 模型旨在提取完整语音段，AA 模型旨在提取浊音段。实际使用中，根据需求选择合适的检测模型。

2.2 初步特征提取

语音数据是一种时序数据。通常，序列的特征可以大致分为两种：时序特征以及局部特征。这里，时序特征建模方式较为有限，而时序特征正是语音信号的一个重要特征：例如基音频率的变化等。本次工作中我们通过循环神经网络进行时序特征的建模。具体地，我们使用 Gated Recurrent Unit(GRU) 作为一个循环单元。局部特征的实现方式比较多样，例如卷积神经网络（CNN），注意力机制（Attention），层级性的网络设计等。这些模型在下文中会更详细地介绍。

特征提取是构建分类模型的首要步骤。有大量的特征可供我们选择：时间域上的特征包括幅值，过零率等；频域上的特征包括基音频率，基于短时傅里叶变换（STFT）的频谱图，Mel 倒谱系数（MFCC），以及它们的差分等。然而，并不是所有的特征都是好的特征。某些特征由于缺少泛化能力，会显著降低模型的性能。另外，使用连续的数值表示这些特征，还是使用参数化的方法（e.g. 二次函数拟合的系数），还是通过量化转换为离散值，对性能都十分关键。由于各个特征的数量级有很大的差别，是否进行放缩、标准化，也是一个容易忽略但是对性能有十分重大影响的要素。

MFCC 特征通过如下的方式提取 [1]。

1) 首先进行端点检测。端点检测选择 AZ 模型。
2) 截取语音段，预加重，分帧并加窗。帧长为 20ms, 跨度为 10ms。窗口函数选择海明窗。
3) 针对每一帧，进行 FFT 变换，取频谱的平方，再取对数。
4) 通过 M 个 Mel 滤波器组进行滤波。本次工作中使用了 26 个滤波器。
5) 进行离散余弦变换。每一帧产生一个 13 维度的特征。DCT 的计算公式为 $C_n = \sum_{k=1}^{M} logx(k)cos(\pi(k-0.5)n/M)$. 其中 $M = 26, N = 1, 2, ..., 13$
6) 可以计算 delta MFCC, 获得一阶和高阶差分特征。本工作最高计算到 2 阶差分。

然而，不同的音频经过 MFCC 计算后幅值的差异比较大。经观察数据发现，0 阶 MFCC 的数值绝对值甚至可以超过 30. 对于这样的数据，直接使用神经网络进行拟合会得到不理想，不稳定的表现。为此，我们以下列步骤使特征趋向于同一量级。针对这一问题，对于整个数据集做归一化并不能得到理想的结果。较好的做法是，针对每一个实例（语音）单独进行

1. 参考了 https://github.com/jameslyons/python_speech_features

一定程度的归一化。然而，盲目的归一化会使我们丢失很多有用的信息。事实证明，以下的操作序列表现较好，能提升大约 2% 的准确率。

1) 对于每个单独实例，计算 0 阶 MFCC 的均值，并从 0 阶 MFCC 的值减去该值
2) 计算 1 阶与 2 阶 MFCC。
3) 对 0 阶 MFCC 进行放缩，使得方差为 1。

得到的 0 阶，1 阶，2 阶 MFCC 特征向量分别为 13 维。经过拼接后，得到一个 39 维的特征向量 x，这个向量就是神经网络的输入。此外，我们还可以将幅值等特征拼接在 x 上。然而，事实证明这些特征是无效的。此外，尽管神经网络允许我们将更大的特征向量作为输入，例如 STFT 获得的语谱图。但是，MFCC 作为一个压缩的时间域-频域表示，相较于 STFT 语谱图仍有很大的优势。MFCC 的倒谱计算与 MEL 滤波器事实上起到了特征去耦合的作用，因此它能提供更优越的特征输入。而且，尽管 MFCC 特征的维度不大，但是我们可以发现即便模型的参数设置地比较大，模型也没有严重过拟合的趋势。

3 深度神经网络模型

本章，我将介绍本工作中我所设计的神经网络模型结构。每个模型的建模都有着明确的目的，设计的模型简洁且符合直觉。具体地，我将介绍建模时间序列的朴素 RNN 模型，建模层次关系与时间序列的 CNN-RNN 模型与层次 RNN 模型，自适应地建模层次关系的自适应 HMRNN 模型。基于手动特征选取的模型将会在下一章节介绍。

3.1 Vanilla GRU 模型

在本节中，我将介绍朴素的 GRU 分类模型。GRU 是一种改进的循环神经网络。由于 GRU 的连续的两个输入之内有一条乘法较少的路径，可以更好地处理矩阵连乘带来的梯度消失和爆炸的问题。除此之外，可选的循环单元还有 Long-short term memory(LSTM) 等，但是 GRU 的运算量相比 LSTM 较小一些，而表现相近。它的计算公式如下。其中，x_t 代表输入的特征序列中第 t 个位置的元素，h_t 是 t 时刻模型的隐状态输出。可以看出 h_t 作为 GRU 单元下一次循环中的输入，这也是循环神经网络的名字由来。h_t 的维度 h 被称为该 GRU 单元的隐层大小。另外，h_0 即初始隐状态为全零向量。

$$
\begin{aligned}
z_t &= \sigma(W_z[h_{t-1}, x_t]) \\
r_t &= \sigma(W_r[h_{t-1}, x_t]) \\
\tilde{h}_t &= tanh(W[r_t * h_{t-1}, x_t]) \\
h_t &= (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t
\end{aligned} \tag{1}
$$

其中，W_z, W_r, W 是可以训练的参数。为了提升模型的表现，一个好的实践是对 GRU 单元的所有参数矩阵进行正交初始化。σ 为某一个激活函数，本工作中选择 tanh 函数作为激活函数。

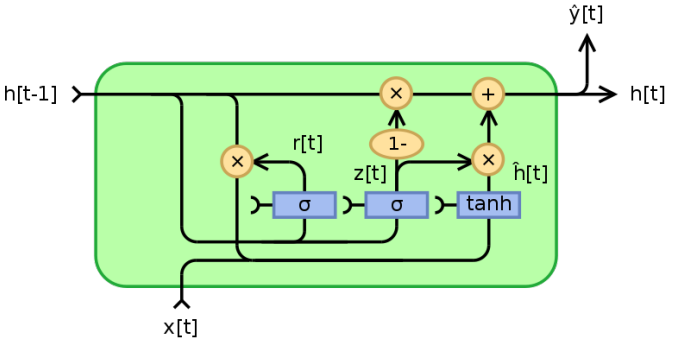

图 1. GRU 循环单元的结构

由于上一时刻的输出作为下一时刻的输，GRU 是建模时间序列的十分理想的模型。我们通常一次性将长度为 L 的一个序列 $x_1, x_2, ..., x_L$ 输入到 GRU 中，并获得长度为 L 的 h 维向量输出 $h_1, h_2, ..., h_L$。该过程简记为：

$$
h_1, h_2 ..., h_L = GRU(x_1, x_2, ..., x_L) \tag{2}
$$

通常，为了进一步提升模型的表现，我们需要使用多层的循环神经网络。本工作中，多层指通过“堆叠”的方式加大神经网络的深度。令 h_t^l 表示第 l 层 GRU 的第 t 个输出。以两层 GRU 网络为例，令 $\mathbf{GRU_1}$ 表示第一层 GRU，$\mathbf{GRU_2}$ 表示第二层 GRU。那么，堆叠构成的两层 GRU 的输入与输出的关系为：

$$
\begin{aligned}
h_1^1, h_2^1, ..., h_L^1 &= \mathbf{GRU_1}(x_1, x_2, ..., x_L) \\
h_1^2, h_2^2, ..., h_L^2 &= \mathbf{GRU_2}(h_1^1, h_2^1, ..., h_L^1) \\
h_1^3, h_2^3, ..., h_L^3 &= \mathbf{GRU_2}(h_1^2, h_2^2, ..., h_L^2)
\end{aligned} \tag{3}
$$

更深层次的 GRU 同理。本工作中 GRU 层数为 3 层，更深层次的网络会有明显的过拟合现象。此外，通常 GRU 等层与层之间可以加入一层 Dropout 层防止过拟合，提升模型表现。本篇工作中若不特殊指明所有的 GRU 网络都是双向结构。一个双向的 GRU 包含两套不同的参数，分别以正向序列与反向序列作为模型的输入。网络最终输出为两个方向的 GRU 对应位置的输出之和。

通过朴素的 GRU Baseline 模型进行分类时，首先将提取的 MFCC 特征序列 $x_1, x_2, ..., x_L$ 并输入到两层的 GRU 中。GRU 的隐层大小为 200 维。之后，我们对得到的长度为 L 的输出向量按照时间维度进行全局平均池化与全局最大池化，得到 200 维的 h_{avg} 与 h_{max} 向量。它们就是我们通过 GRU 提取的最终的特征。

$$
h_{avg} = avgpool(h_1^3, h_2^3, ..., h_L^3) \tag{4}
$$

$$
h_{max} = maxpool(h_1^3, h_2^3, ..., h_L^3) \tag{5}
$$

对于较短的序列，使用最后一个隐藏层作为最终提取的特征也是一个很好的选择。然而，对于较长的序列，这种做法效果不如全局平均池化与全局最大池化的特征提取。

最后，我们将提取的特征输入进全连接层，并通过 softmax 函数对得分进行归一化，求出语音段属于某个词的概率 $P(w|X)$。

$$P(w|X) = softmax(W_{dense}[h_{avg}, h_{max}]) \tag{6}$$

模型使用交叉熵函数作为损失函数。在测试时，模型输出概率最高的下标对应的单词。

尽管朴素的 GRU 模型设计十分简单，在实验结果一章节能看到，它能获得非常好的效果。

3.2 CNN-RNN 局部-时序模型

朴素的 GRU 模型能够很好地建模语音数据的时序性，然而，它并不能很好的处理语音的局部特征。上述的朴素的 GRU 模型中，模型的全连接层的输入有两种选择。第一种是本次工作使用的全局池化，然而，由于序列较长，大量的信息会受到损失。第二种是选择最后一个隐藏层，但这也不能很好地刻画局部特征。

显而易见的是，语音信号的局部性是十分重要的特征。一段语音可以划分为多 “音素” 作为基本组成部分，如果对这种局部性的音素特征能进行合适建模的建模，可以预期效果会有比较大的提升。卷积神经网络（CNN）被证实可以有效地提取局部特征。因此，我们提出 CNN-RNN 局部-时序模型。

CNN 可以使用于不同维度的数据。对于 STFT 语谱图特征，使用时间-频率二维卷积效果可能较为理想。然而，正如一些工作所提到的，对于维度不大的 MFCC 系数特征仅对时间域进行仅对时间维度卷积效果更好。一个 CNN 网络层输入由 M 个输如信道组成，输出由和 N 个输出信道组成。CNN 网络的核心是卷积核。在运算的过程中，卷积核在输入上滑动，同时计算窗口内输入与卷积核的卷积。某一维度卷积核的大小 k, 以及沿着某一维度的步长 s 都是可以设置的参数。如上文所讲，本次工作中仅针对时间域进行卷积。卷积核的长度 k 为 11 帧，步长为 5 帧。之后，对输出进行滑动窗口平均池化，池化的窗口宽度为 10. 最后，CNN 的输出作为单层 GRU 的输入，再进行分类。这时，GRU 的输入序列长度已经大幅度减小了。

然而，事实证明，CNN-RNN 不如接下来所提到的局部-时序模型的实现方式。

3.3 Hierarchical RNN 层次-时序模型

层次 RNN（HRNN, Hierarchical RNN） [1] 是一种层次化的时间序列建模方式，多用于建模文本数据，这是因为文本数据包含天然的层次结构：字符，单词，语句等。对于语音信号，尽管各个音素使语音数据具有很好的层次结构，但是边界并不能很容易地提取。然而，我们仍然可以大致估计组成语音片段的音素的层次结构的量级。

同 Vanilla RNN 模型一致，Hierarchical RNN 模型由 3 层 GRU 组成，其中第三层具有层次结构，分别记为 GRU_1,GRU_2,GRU_{hr}。前两层等同于朴素的 GRU 模型中的双层 GRU。$\mathbf{x}$ 为输入的 MFCC 特征序列。第三层 GRU 对第二层 GRU 的输出进行等间隔采样，间隔 N_t 需要人为估计。本工作中，我发现以 5 帧 (帧移为 0.01 秒) 为间隔进行采样是一个很好的选择。这样，对于一段长为 0.8 秒的语音，会产生一个长度为 16 的采样序列，这是一个长度十分合适的一个序列。模型对采样间隔 N_t 不是很敏感，这是因为前两层 GRU 本身就有建模时间序列的功能，一定程度上起到了对齐的作用。具体地，特征提取的公式如下。

$$\begin{aligned}
h_1^1, h_2^1, ..., h_L^1 &= \mathbf{GRU_1}(x_1, x_2, ..., x_L) \\
h_1^2, h_2^2, ..., h_L^2 &= \mathbf{GRU_2}(h_1^1, h_2^1, ..., h_L^1) \\
h_1^3, h_2^3, ..., h_{\frac{L}{N_t}}^3 &= \mathbf{GRU_{hr}}(h_1^2, h_{1+N_t}^2, ..., h_{1+(k-1)N_t}^2)
\end{aligned} \tag{7}$$

接下来，仅对最后一层 GRU 的输出进行全局池化，并输入进全连接层并分类，求得语音段属于某一类别的概率。

$$h_{avg} = avgpool(h_1^3, h_2^3, ..., h_{\frac{L}{N_t}}^3) \tag{8}$$

$$h_{max} = maxpool(h_1^3, h_2^3, ..., h_{\frac{L}{N_t}}^3) \tag{9}$$

$$P(w|X) = softmax(W_{dense}[h_{avg}, h_{max}]) \tag{10}$$

3.4 Hierarchical MultiScale RNN 自适应层次-时序模型

层次 RNN 模型的一个缺陷是层次的划分是固定的。就比如上述模型中，我们以 5 个帧为层次进行划分。显然，这种固定的层次划分模式可能会产生一些误差。试想，我们提出层次 RNN 模型的目的是使模型能更好地从局部性的音素特征提取信息。既然固定层次可能产生较大误差，那么我们能否将其设计为自适应音素层次的结构呢？如果数据集有音素划分的标注，那么这个任务就是琐碎的。然而，获取此类数据代价巨大。为此，我们重新将目光放在 RNN 网络结构的设计上。Hierarchical Multiscale RNN(HMRNN) [2] 的提出就是针对这一目标提出的。HMRNN 起初用于处理文本数据：在处理过程中自动对若干单词组成的短语进行划分，借此在 “单词” 和 “句子” 中划分出了一个新的层级，提升了多项任务的表现。“音素” 的划分与 “短语” 的划分有一个共同的性质，就是长度并非固定。

我们结合 HMRNN 结构与 LSTM 作为循环神经网络计算单元。由于实现细节较为繁琐，本工作使用作者发布的 HM-LSTM 单个运算单元的实现，但对结构做了一些修改。其中，LSTM 单元是一种不同于 GRU 单元的较早提出的神经网络单元结构。相比 GRU 单元，LSTM 单元额外输出 “单元状态 (cell state)”，这也是 LSTM 单元参数量要大于 GRU 单元参

数量的原因。通常，单元状态输出不用作最终提取的特征，而仅用于将信息传递到下一步的输入中。相比普通的 LSTM 单元，HM-LSTM 计算单元额外输出一个控制变量 z. 对于一个 l 层次结构的 HM-LSTM 网络，输入输出的关系可以用下面的式子描述。

$$h_t^{(l)}, c_t^{(l)}, z_t^{(l)} = f(c_{t-1}^{(l)}, h_{t-1}^{(l)}, z_{t-1}^{(l)}, z_t^{(l)}, x_t) \tag{11}$$

其中，$h_t^{(l)}$, $c_t^{(l)}$ 是第 l 层第 t 个元素的 LSTM 隐状态。$z_t^{(l)}$ 代表一个控制变量。本工作中选用了 2 层的 HM-LSTM 结构（即 $l \in 1, 2$），其中，上层 LSTM 体现层次结构。上层的 LSTM 的行为受 $z_t^{(1)}$ 与 $z_{t-1}^{(2)}$ 的控制。遵从原文的设计，状态 c_t 有如下三种行为：

- **更新**。此时，LSTM 正常输入旧状态，输出新状态。
- **复制**。即 LSTM 的状态保持不变。
- **归零**。LSTM 将自身的输出归零。

状态兼输出 h_t 有如下两种行为：

- **更新**。LSTM 正常输出。
- **复制**。LSTM 复制之前的输出。

其中，更新的触发条件为 $z_t^{(1)} = 1 \wedge z_{t-1}^{(2)} = 0$，复制的触发条件为 $z_t^{(1)} = 0 \wedge z_{t-1}^{(2)} = 0$, 归零的触发条件为 $z_{t-1}^{(2)} = 1$。可以看出 z 是模型输出的决策变量。这个变量对模型的控制作用是离散的，不能直接通过反向传播来训练。一种做法是将 z_t 视为决策序列，并运用强化学习的算法，但是通常这种算法运行效率低下且效果不稳定。我们按照在 HMRNN 文章中提出的策略，套用 slop-annealing 技巧。具体地，在模型的前向计算过程中，模型实际输出介于 0 到 1 的输出 $\tilde{z}_t$，随后离散化为 z_t 即：

$$z_t = f_{bound}(\tilde{z}_t) = \begin{cases} 1, & \tilde{z}_t > 0.5 \\ 0, & otherwise \end{cases} \tag{12}$$

当梯度反向传播至无法计算梯度的 z_t 时，我们使用 hard-sigm 函数的梯度近似这一离散操作的梯度。

$$sigm(x) = max(0, min(1, \frac{ax+1}{2})) \tag{13}$$

其中，a 是一个可以设定的参数。a 越大，坡度越陡峭（这也是 slope annealing 名称的由来）。根据作者给出的建议，训练起初，a 设定为较小的值（如 1），随着模型的收敛，不断地增加 a。

自适应层次-时序模型的模型框架如下。模型的前两层与 HRNN 相同，是两层双向 GRU，以 MFCC 以及 1 阶，2 阶差分作为输入，提取出 200 维的特征序列。之后，将该特征序列作为 HM-LSTM 的输入。我们使用双向 GRU 提取出的特征的全局池化，平均池化，以及 HM-LSTM 的最后一个隐层输出作为全连接层的输入。全连接层的输入时根据实验得到的最佳选项。与 Vanilla RNN，HRNN 的三层网络结构不同的是，HMRNN 模型的 RNN 结构总计有四层：分别是前两层的 GRU，以及 HM-LSTM 的两层。实验发现 HMRNN 在这一设置下表现最佳，但是 Vanilla RNN 与 HRNN 均在 RNN 层数为 4 层发生过拟合，表现甚至不如 3 层。因此，我们认为这样的参数选择比较是合理的。

具体地，HMRNN 模型的设计可以表示为如下。

$$\begin{aligned}
&h_1^1, h_2^1, ..., h_L^1 = \mathbf{GRU_1}(x_1, x_2, ..., x_L) \\
&h_1^2, h_2^2, ..., h_L^2 = \mathbf{GRU_2}(h_1^1, h_2^1, ..., h_L^1) \\
&h_1^3, h_2^3, ..., h_L^3 = \mathbf{HM-LSTM}(h_1^2, h_2^2, ..., h_L^2) \\
&h_{avg} = avgpool(h_1^2, h_2^2, ..., h_L^2) \\
&h_{max} = maxpool(h_1^2, h_2^2, ..., h_L^2) \\
&P(w|X) = softmax(W_{dense}[h_{avg}, h_{max}, h_L^3])
\end{aligned} \tag{14}$$

3.5 其他模型

除上文提及的模型之外，我还尝试了加入 Attention 机制的层次 RNN 模型。Attention 机制可以理解为动态地对 RNN 输出序列分配权重，而不是使用平均池化。然而，事实证明这没有带来更好地效果。此外，我还尝试了基于 Attention 的无需 RNN 的 Transformer 编码器，旨在挖掘语音信号潜在的“长程依赖”特性。然而，事实证明这一改进是无效的。

4 声调判别模型

基于深度神经网络的模型已经可以获得令人满意的结果。但是，它们倾向于对特定的几个中文单词混淆。这些词大多有相同的拼音，不同点仅仅是声调。例如“北京”与“背景”以及“语音”与“余音”：根据人们的直觉与常识，在“余音”这一词语中，首个单字为二声，因此，声调——也就是基音频率应该显示递增的趋势。在“语音”这一词中，首个单字为三声，基音频率的改变理应与事实已经证明了仅依靠模型学习声调信息并不可靠，我们需要更加显式地手动构造特征，并对这些易混淆的词语进行分类。

本章以模型最容易混淆的“语音”与“余音”以及“北京”与“背景”为出发点进行说明。本章的重点在于提取基音频率的方法。提取一些基音频率特征后，分类的任务是很容易的，我们只需使用简单的线性分类器就能得到很好的效果。

4.1 倒谱法

浊音信号的产生可以用线性的激励模型 G 与声道模型 V 进行建模（我们忽略辐射模型）。由于激励模型与声道模型是串联关系，时域的信号实际上经过了卷积。我们可以通过同态分析的技术，也就是倒谱法，在时域上分离激励模型与声道模型。为了计算基音频率，我们只需声门激励信号的信息。经过倒谱变换后，信号的快变分量（声道相应）与慢变分量（声门激励）在时域上分离，在每个基因周期位置形成一个峰。

倒谱法获得基音频率的过程如下：

1) 对语音进行预加重。事实证明这一步对于检测的稳定性有很大的影响。预加重系数为 0.97.
2) 对语音片段进行下采样，采样为 10kHz.
3) 分帧。人声的基音频率为 50Hz-500Hz，即单个周期在 20ms 到 2ms 之间。为了保险地至少包含两个完整的基因周期，帧长被设定为 51.2ms. 这相当于每帧 512 个点。帧移设置为 10ms.
4) 通过之前提到的 AA 模型，精确地对浊音片段进行端点检测。由于声调判别模型仅用于双字的中文单词的识别，我们额外检测每个单字的区间。检测通过识别幅值的谷点来完成。
5) 通过线性相位低通滤波器进行滤波，截止频率 900Hz. 滤波使用汉明窗。
6) 计算倒谱：先对信号进行 512 点 FFT，并取幅值。再对信号进行 512 点 IFFT，再取幅值。
7) 对每 3 个帧的倒谱信号幅值取平均，进行光滑。
8) 通过一定的策略，提取基音频率信息。

最后一步的最简单的实现方式是对每一帧的倒谱取第一个峰值点，作为基音周期。然而，如果倒谱信号的一个峰值不明显，很可能出现半频错误（倍频错误不易发生）。注意，如果一段语音信号全程半频或倍频，对分类造成的影响并不大，因为我们重点关心基音频率的变化而非绝对值。基音频率的绝对值意义不大，因为它本身就因人而异。然而，如果提取的基音频率持续地在半频、正常、倍频见跳变，这会对算法性能有比较大的阻碍。有两种额外策略可以减轻这种现象。第一种方式基于设定的半频与倍频容错阈值 C_1 与 C_2，通过贪心的方式平滑化。

1) 检测第一个峰值。倒谱非常不平滑，因此不能简单以第一个局部极大（导数为 0）作为判断峰值的标准。因此，我将采取以下的方法提取峰值：首先对各个帧重新打分，分数为从该帧向两端延伸，直到遇到倒谱幅值超过该帧延伸的帧数。记这些分数为 $score_{fix}$。取分数最大值对应的帧。
2) 从第二个帧开始，如果该帧提取的基音频率落于 2 倍的上一帧基音频率的 C-邻域，即绝对值相差小于 C，那么将该帧的幅值修改为 2 倍。C 设定为 25Hz。

另一种方式是通过动态规划的方式平滑化，算法非常类似于求解 HMM 所用的维特比算法。给定一个由 $score_{fix}$ 组成的，长度为帧数的序列，找到一条前向路径，使得积累的 $score_{fix}$ 累积最大，但是如果选择频率上相差太远的点，会有一定惩罚。这一方法在其他文献中使用过，但是经过实验后发现，这一方法相对于前一种基于容错阈值的方法效率低很多 (改算法复杂度为 $O(TN^2)$，容错阈值法复杂度仅为 $O(TN)$，其中 T 为帧数，N 为单个帧内的采样点数)，且没有明显的效果提升。

得到第一个峰值对应的点下标后，基音频率通过如下公式计算。

$$f = \frac{f_s}{N} = \frac{10000}{N} \tag{15}$$

其中，f_s 是采样频率，N 是采样点数。我们可以估算当 $N = 50$，也就是基音频率为 200Hz 左右时，该方法分析基音频率的分辨率。

$$f_0 = 200 - \frac{10000}{51} = 3.92Hz \tag{16}$$

这一分辨率已经是比较理想的水平。然而，由附图可以发现倒谱法获得的峰值并不明显。我们尝试使用另外一种方法：自相关法提取基音频率。

4.2 自相关法

自相关法是一种提取基音频率的常用方法。它不需要复杂的频谱分析，所有操作在时域中即可完成。给定一个帧，我们通过下式计算帧内的自相关值。

$$R(t) = \sum_{i=0}^{N-t-1} x(i)x(i+t) \tag{17}$$

直观上，当 t 恰好为基因周期的整数倍时，$x(t)$ 与 $x(t+i)$ 倾向于重合，因此自相关值很高。因此，自相关法提取基音频率的方式与倒谱法类似，都是寻找第一个峰值点，该峰值点的偏移对应基音周期。自相关法提取基音频率的算法如下：

1) 对语音进行预加重，下采样，分帧，端点检测，滤波等操作与倒谱法相同。
2) 使用削波函数，平滑掉幅值较低的采样点。设定阈值为该帧幅值的中位数。若 $|x(t)| < median$，那么 $x(t)$ 被平滑为 0. 否则，将 x(t) 的幅值减少 median。
3) 计算 $t \in [20, 200]$ 的自相关函数。这对应于基因周期在 2ms 到 20ms 之间，即基音频率 50Hz 到 500Hz 之间。
4) 从自相关值通过一定的策略，提取基音频率信息。

与倒谱法得到的幅值倒谱相比，自相关函数的谱图要光滑许多。因此，我们可以简单通过找到最大值点所对应的偏移量 $t*framelen$ 作为基音周期。在基音频率分辨率层面上，倒谱法与自相关法是等价的。然而，由于自相关谱更为光滑，因此提取峰值更为方便。值得一提的是，倒谱法在计算复杂度上优于自相关法。倒谱法计算量很小：只需计算一次 FFT，一次 IFFT 与其他线性时间操作。算法复杂度为 $O(TNlogN)$。自相关函数计算的过程需要平方级的复杂度，且要对多个 t 重复计算 $R(t)$，时间复杂度为 $O(TCN^2)$，其中 T 为帧数，N 为单个帧的点数，C 为 $R(t)$ 中 t 的取值区间大小。实验过程中可以明显感受到两种方法速度的差异。

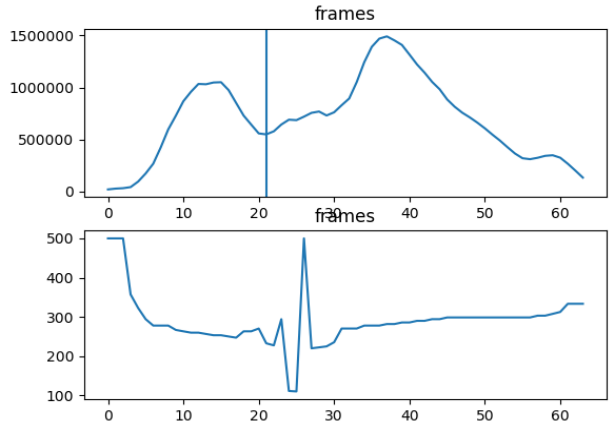

图 2. “语音”对应的语音段幅度谱与基音频率谱。上图的竖线表示检测到的双子词的分割点。

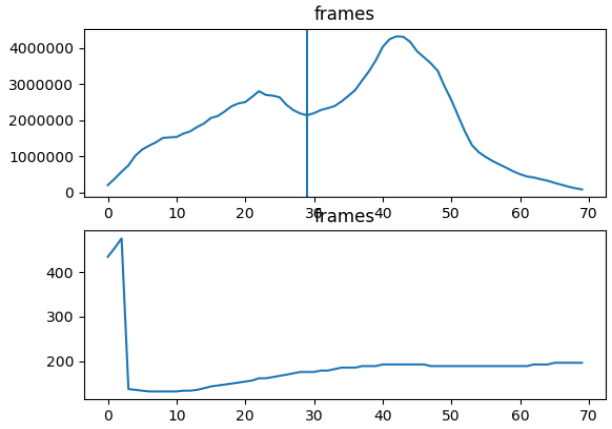

图 3. “余音”对应的语音段幅度谱与基音频率谱。上图的竖线表示检测到的双子词的分割点。可以看到基音频率不断上升

4.3 特征提取

提取了语音的基音频率信息后，我并不打算像之前一样通过 RNN 等模型进行训练，而是手动构造特征。首先，通过一定规则提取两个字的基音周期的“平稳序列”，记为 S_1 与 S_2。可以构思以下特征：

1) 通过一次函数拟合 S_1 与 S_2，取斜率。
2) 通过二次函数拟合 S_1 与 S_2, 取二次项系数 a。
3) 两段的最高基音频率之差.

直观来看，斜率的确是最重要的。比如，“语音”一词中，“语”的基音频率应该较为平稳，或者有下降的趋势；而“余音”一词的“余”中，基音频率应有上升的趋势。在“北京”与“背景”两个词同理。我们可以选择一个常用的线性分类模型进行判别，本次实验中我选择线性 SVM 进行分类。

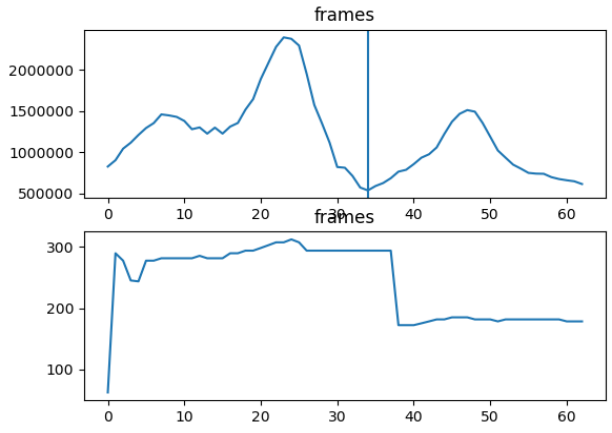

图 4. 使用自相关法提取的前半部分发生半频错误的一段语音。如上文所述，自相关法容易发生倍频错误。然而，这并不影响判决，因为在单字中基音频率变化趋势是正确的

4.4 模型组合

设计基于基音频率变化的判别模型旨在协助上一章所提到的强有力的神经网络判别模型。在验证集中，神经网络的分类错误几乎都在“北京”与“背景”，“语音”与“余音”这两个词上。因此，我使用模型组合（model ensemble）技术，对神经网络模型与基于基音频率的模型进行组合。

SVM 不能输出概率。因此，我们通过以下方式进行模型组合：当神经网络输出结果为语音或余音时，若概率小于 80%，则调用 SVM 分类器进行分类，结果以 SVM 分类器的结果为准。对于北京与背景同理。

5 试验：通过 Adversarial Distilling 去除说话人不同的影响

由于训练集与验证集、测试集之间的说话人没有重叠，因此，说话人信息对于我们的模型是一种干扰。比如说，不同的说话者有不同的基音频率；或者，由于不同的说话者有各自的口音，声调也会有轻微的不同，而这似乎并不容易用手动特征工程的方式进行建模。而通过神经网络提取的特征实际上既包含了单词信息，也包含了说话人信息。我们很希望通过某种方式，将学习到的特征进行“提纯（distill）”[4]，使模型更加专注于建模不受说话人影响的“单词信息”。我们预期这将会增加模型的鲁棒性。

我们对“学习单词相关的，说话人无关”的学习任务进行显式的建模。我们额外构造一个引入 Adversarial Training 思想的分类器：使模型通过提取的特征，最小化单词分类错误率，同时最大化说话人分类错误率。这里，说话人分类器相当于一个“对抗网络 (Adversarial Network)”中的判别器，试图阻止模型提取出与说话人强相关的特征。通过查阅论文后我发现语音识别领域似乎没有太相关的工作。已知这一方法在自然语言处理领域，计算机视觉领域的 domain adaptation 领域有一定的应用，并取得了较好的成绩。事实上，在 2014

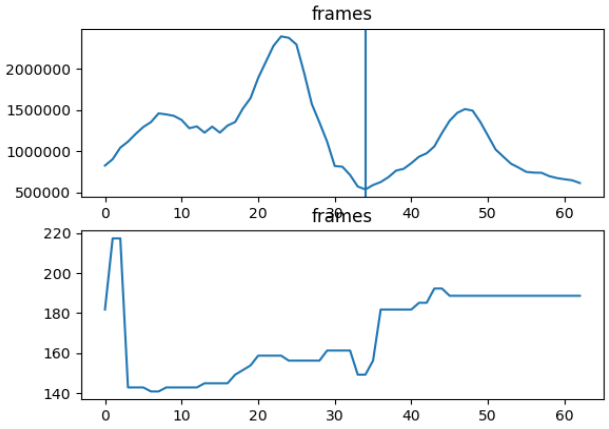

图 5. 使用倒谱法提取的与图 3相同语音的基音频率。倒谱法不容易发生倍频错误，但是半频错误相对较多。

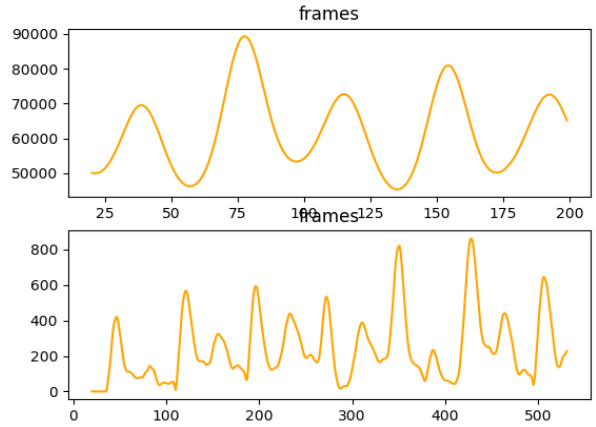

图 6. 单帧的自相关谱与幅值谱。可以看到自相关谱比较光滑。

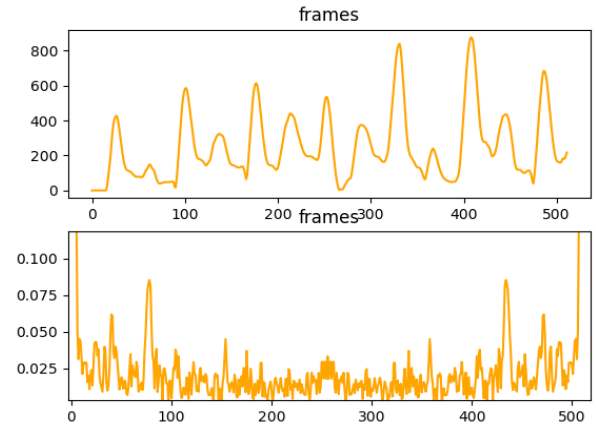

图 7. 单帧的幅值谱与倒谱。倒谱不如自相关谱光滑，但是峰值也十分明显。

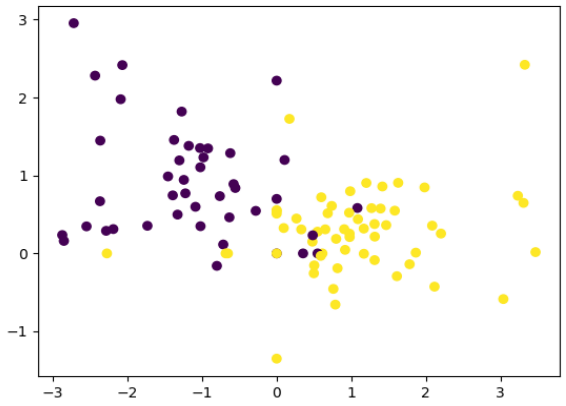

图 8. 通过自相关法提取的基音频率特征的斜率信息进行“语音”与“语音”分类，画出约 100 个实例的散点图。可以发现点簇被明显的划分。其中，紫色点表示“语音”，黄色点表示“余音”。x 轴表示第一个词的基音频率斜率，y 轴表示第二个词的基音频率斜率。这个图的结果是很容易理解的，比如，“语”的音调普遍是下降趋势，而“余”的音调普遍是提高趋势

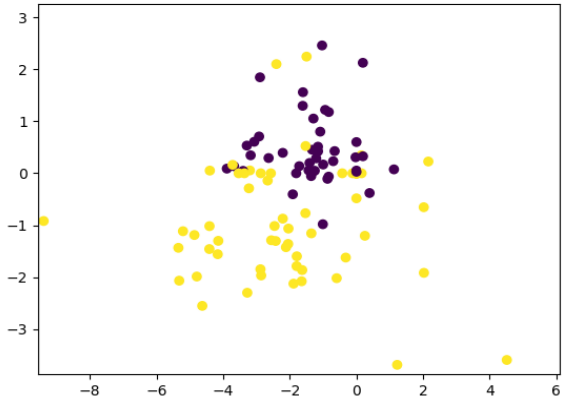

图 9. 通过自相关法提取的基音频率特征的斜率信息进行“北京”与“背景”的分类，画出散点图。其中，紫色点表示“北京”，黄色点表示“背景”。x 轴表示第一个词的基音频率斜率，y 轴表示第二个词的基音频率斜率。“北”的音调不确定，“背”的音调下降。

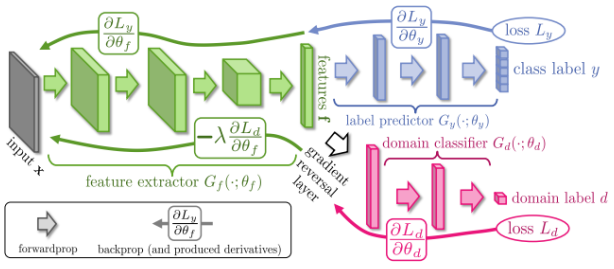

图 10. [3] 中关于模型的图示，也可以作为本工作的框架图示。domain classifer 在本工作中替换为 speaker classifer

年这种方法就已经被提出 [3], 目前仍有许多人在改进，并应用于多种其他任务。

我们使用 3 层 Vanilla GRU 作为 baseline 模型并引入 [3] 的结构。回想该网络提取的特征为：

$$h_{avg} = avgpool(h_1^3, h_2^3, ..., h_L^3) \tag{18}$$

$$h_{max} = maxpool(h_1^3, h_2^3, ..., h_L^3) \tag{19}$$

之后，将特征全连接层，并标准化特征得到，得到各个词对应的概率。

$$P(w|X) = softmax(W_1[h_{avg}, h_{max}]) \tag{20}$$

我们将特征输入进另一个全连接层进行说话人分类，得到各个说话人的概率。

$$P(S|X) = softmax(W_2[h_{avg}, h_{max}]) \tag{21}$$

将特征提取网络记为 $\mathcal{F}$，单词分类器记为 $\mathcal{D}_1$，说话人分类器记为 $\mathcal{D}_2$。D_1 与 D_2 专注于各自的分类任务。

$$\begin{aligned} \mathcal{L}_{word} &= -logp_{word} \\ \mathcal{L}_{spk} &= -logp_{spk} \\ \nabla\mathcal{D}_1 &= \nabla\mathcal{L}_{word} \\ \nabla\mathcal{D}_2 &= \nabla\alpha\mathcal{L}_{spk} \end{aligned} \tag{22}$$

其中 α 是一个权衡因子，设定为 0.1。特征提取网络 $\mathcal{F}$ 的学习目标为最小化单词分类错误率，最大化说话人识别错误率。因此，$\mathcal{F}$ 的梯度方向与 $\mathcal{D}_2$ 相反。

$$\nabla\mathcal{F} = \nabla\mathcal{L}_{word} - \nabla\alpha\mathcal{L}_{spk} \tag{23}$$

梯度取反的实现上只需定义一种函数，前向计算时表现为 Identical Function，后向传播时将梯度取反即可。整个模型可以端到端学习。

本次实验中，上述模型并没有取得预期的效果，这可能是由于数据集较小，且模型的错误主要不是来自说话人不同带来的噪音，而是其他因素（例如，声调分辨不清）。然而，我坚信以这种对抗网络的方式对特征进行提纯的思路是正确的，因为我们用损失函数与梯度显示地定义了学习目标，即学习与另一个任务无关的特征。而且，这一框架具有很好的通用性。我相信这一模型在更大的数据集，以及更精细的参数设置后会取得很好的效果。

6 实验设置

本次实验仅适用 2018 年数字信号处理课程同学们录制的音频作为训练与验证数据。模型的最终测试的说话人的音频不存在于数据集。整个数据集被分割为 4:1，分别为训练集与验证集。不同集合之间的说话人不会重叠，这会使得验证与测试的场景较为相近。训练集包含的说话人与验证集包含的说话人

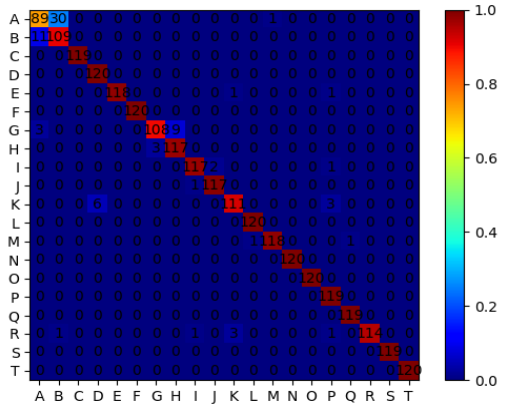

图 11. 神经网络分类器的典型的混淆矩阵，可以看到神经网络的分类错误集中在第一个词（语音）与第二个词（余音）之间，其次是第六个词与第七个词（北京，背景）。从对角线元素的颜色可以估计准确率。“语音”一次的准确率仅为大约 0.7。然而，神经网络模型在其他词上的准确率都接近完美。

通过随机的方式进行抽取。音频的采样率各不相同，有些是 44.1KHz，还有些是 48KHz. 音频以双声道的波形文件（.wav）保存，但是两个声道的数据是完全相同的，因此实际实验中只使用一个声道的数据。

本实验的代码通过 Python 编写，数值处理与机器学习部分使用了 numpy, scipy, sklearn 框架；绘图使用了 matplotlib 框架；神经网络部分使用了 pytorch 框架，除了 HMRNN 部分在开源实现基础上修改 [2]，其它神经网络部件均自行编写。

SVM 训练过程中我们使用 RBF kernel. 神经网络训练过程中，我对所有模型使用同一个随机种子，批大小为 32，学习率为 0.001，使用 Adam 优化器。每经过一个 epoch，学习率衰减为原来的 0.8 倍，如果在验证集上准确率下降，那么衰减为原来的 0.5 倍。累积 3 次发生验证集准确率下降后训练结束。

除了神经网络之外，我们还比较以下模型的表现。

1) PitchSVM01。表示通过基音频率特征训练的“语音”与“余音”的分类器。
2) PitchSVM67。表示通过基音频率特征训练的“北京”与“背景”的分类器。

后缀名表示实现方法。例如，PitchSVM01-Cesptral 表示使用倒谱法提取的基音频率特征并进行分类的模型。

7 实验结果

表 1给出了神经网络模型的分类准确率。惊人的是，纯 RNN 的效果已经十分优秀，达到了 96.3% 的准确率，而在 RNN 上简单增加层次的 HRNN 与 CNN-RNN 模型准确率为 96.1%

2. https://github.com/HanqingLu/MultiscaleRNN

Vanilla RNN	HRNN	CNN-RNN	HMRNN	Adversarial	Ensemble HMRNN + PitchSVM(AutoCorrelation)
0.963	0.961	0.949	0.966	0.959	**0.971**

表 1
神经网络模型的准确率

PitchSVM01-AutoCorrelation	PitchSVM67-AutoCorrelation	PitchSVM01-Cesptral	PitchSVM67-Cesptral
0.887	0.825	0.896	0.858

表 2
独立基于基音频率特征的 SVM 模型准确率

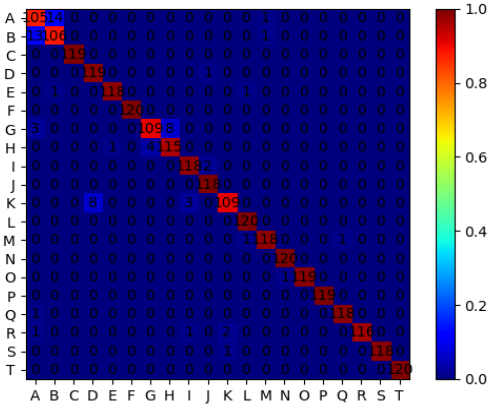

图 12. Ensemble 模型的表现，可以看到神经网络的分类弱点被弥补，在语音与余音之间区分明显

与 94.9%，并没有提升准确率。从模型的设计可以找出一丝端倪：在 HRNN 与 RNN 模型中，GRU 总计都有三层。然而，HRNN 模型与 RNN 模型的区别仅仅是第三层的输入是对第二层的采样。当底层模型输入非常长的时候这种采样时有利的。然而，在本工作中，输入序列长度大约为 80 帧，这或许对于一个 RNN 来讲并不是一个很大的数目，因此也能很好的建模。因此，没有显式建模层次结构的 RNN 效果可能更好。但是，Hierarchical Multiscale RNN 在准确率上的确得到了一定的提升，达到了 96.6%，是表现最好的独立模型。图 11给出了 HMRNN 模型的混淆矩阵热度图，显示了各个词分类的准确率信息。

表 2给出了基于基音频率的二分类模型的准确率，可以看到相比神经网络模型，这些二分类的模型在特定的词之间可以达到更高的准确率。我们还可以看到倒谱法优于自相关法。

表 1还给出了组合模型的准确率。我们可以看到组合模型的准确率又有了较大的提升，达到了 97.1%。图 12给出了组合模型的混淆矩阵热度图，可以看到在“语音”与“余音”之间的分类错误率得到了显著的下降。

8 相关工作

语音识别是一个历史悠久的研究课题，人们已经进行了大量的探索，目前的语音识别技术也已经趋于成熟，大多数最新的模型也投入商业使用。[5] 引入多层感知机 (MLP) 与 HMM 的混合模型用于长文本的识别，是一个开创性的工作。[6] 引入 RNN 模型进行语音识别。[7] 引入双向 LSTM 结构进行长语音识别。此后，神经网络开始广泛用于语音识别，产生了非常多样的模型。然而尽管神经网络对时间序列建模十分强大，他们仍然离不开它们的基石，即数字信号处理相关的技术。动态时间归正的思想，降噪滤波器的思想被神经网络显式或隐式地建摸。[8] 引入注意力机制对语音进行建模。[9] 引入了对语音识别提取的特征进行提纯的思想。[10] 引入对抗生成网络（GAN）的思想搭建了语音识别模型。[4] 通过变分推断进行了语音识别上的 domain adaptation，这与本工作的思路较为相近。我们容易看到，近年的研究热点是如何使用神经网络建模人类显而易见的特征，这一过程中又不断探索合适的网络结构。

9 结论

本工作中，我使用了特征工程与深度神经网络的方式完成了中英文孤立词识别的任务，得到了很好的效果。我对多种端点检测方式与基音频率提取方式进行了实现与研究，并对比了他们多方面的表现。我应用并设计了多种网络结构，旨在对语音进行最优的建模，进行了可靠的对比实验。此外，我还对对抗式特征提纯的模型进行了探索。最后，我将以上的模型进行组合构建分类器，达成了 97.1% 的准确率。

作为未来展望，我希望可以将 HMRNN 用于连续语音识别的任务。由于 HMRNN 的作用相当于自动的端点分割，想必在连续语音识别任务中会有更突出的表现。此外，对于前文提到的对抗式特征提纯技术，想必在更大的数据集会体现出它的优势。

参考文献

[1] R. Lin, S. Liu, M. Yang, M. Li, M. Zhou, and S. Li, “Hierarchical recurrent neural network for document modeling,” in *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 899–907, 2015.

[2] J. Chung, S. Ahn, and Y. Bengio, "Hierarchical multiscale recurrent neural networks," *arXiv preprint arXiv:1609.01704*, 2016.

[3] Y. Ganin and V. Lempitsky, "Unsupervised domain adaptation by backpropagation," *arXiv preprint arXiv:1409.7495*, 2014.

[4] W.-N. Hsu, Y. Zhang, and J. Glass, "Unsupervised domain adaptation for robust speech recognition via variational autoencoder-based data augmentation," *arXiv preprint arXiv:1707.06265*, 2017.

[5] N. Morgan and H. Bourlard, "Continuous speech recognition using multilayer perceptrons with hidden markov models," in *Acoustics, Speech, and Signal Processing, 1990. ICASSP-90., 1990 International Conference on*, pp. 413–416, IEEE, 1990.

[6] T. Robinson, "A real-time recurrent error propagation network word recognition system," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, vol. 1, pp. 617–620, IEEE, 1992.

[7] A. Graves and N. Jaitly, "Towards end-to-end speech recognition with recurrent neural networks," in *International Conference on Machine Learning*, pp. 1764–1772, 2014.

[8] J. K. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in neural information processing systems*, pp. 577–585, 2015.

[9] K. Markov and T. Matsui, "Robust speech recognition using generalized distillation framework," in *INTERSPEECH*, 2016.

[10] A. Sriram, H. Jun, Y. Gaur, and S. Satheesh, "Robust speech recognition using generative adversarial networks," *arXiv preprint arXiv:1711.01567*, 2017.