Project 1：基于 BiLSTM 和 MFCC 特征的简单语音端点检测算法

519030910352 郭奕玮

摘要：本次 Project 基于语音信号梅尔倒谱系数（MFCC）和能量特征，用双向长短时记忆神经网络（BiLSTM）训练了一个语音端点检测模型 VADnet，在验证集上达到了 0.9956 的 AUC 和 0.9749 的 VACC。实验发现 VADnet 只需要少量数据就可以训练得到很好的效果，且预测结果几乎不需要进行后处理，是一个易用而强大的模型。代码已上传于`https://github.com/cantabile-kwok/VAD-LSTM`。

1. 数据预处理及特征提取

Task1 中已对时域的能量和过零率进行了详尽分析，而本次 Task2 选用频域的 MFCC 特征，以期挖掘语音信号更深层次的信息。MFCC（Mel Frequency Cepstral Coefficient）是经过梅尔域转换后的 Fbank，即对原信号做 DFT 之后，加梅尔域的滤波器组，得到 Mel Spectrogram，然后取幅值对数、做 DCT 得到 MFCC 系数。本实验使用 Python 的 spafe 库进行 MFCC 的提取，指定维度为 13 维，默认进行系数为 0.97 的预加重，加 Hamming 窗，进行 512 点 FFT，并进行归一化，以消除不同 Speaker 和录音环境的影响。此外，所有音频信号利用 spafe.preprocessing 工具进行分帧。

利用 spafe 中的可视化工具，对其中一个样本进行 MFCC 后的结果可视化如图1所示。

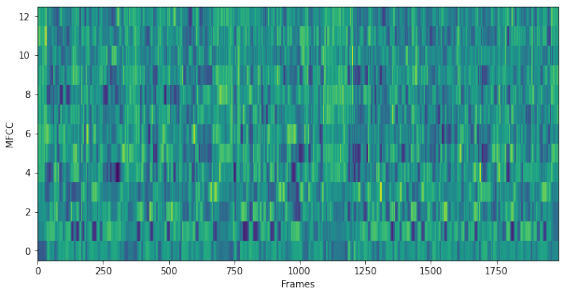

图 1: dev/ 1154-129975-0015.wav *MFCC* 特征提取结果。颜色越浅表示值越大。

由于本实验采用双向 LSTM 进行模型构建，具有强大的捕捉前后文信息的能力，故不计算一阶和二阶差分 MFCC，以加快训练速度。

此外，考虑到能量依然是划分语音和静音帧的一个有效依据，本实验仍然提取能量作为其中一个特征。因此综上，对于任意音频文件，经过特征提取后得到形状为（N，14）的矩阵，其中第一维代表帧数。

2. 算法描述

RNN（Recurrent Neural Network）是一种可以处理变长序列输入的深度神经网络模型，自提出以来 RNN 已有许多变体，LSTM、GRU 是其中最为著名的几种。LSTM（Long Short-term Memory）和 GRU（Gated Recurrent Unit）通过巧妙的门控机制设计解决了传统 RNN 难以长期记忆的问题，增强了上下文建模能力。目前 LSTM 已被广泛用于 NLP、ASR、TTS 等各种序列相关机器学习任务中并取得了出色成果。BiLSTM 指双向 LSTM，即每一个时间戳的输入能同时利用到该时间前后的信息。其特性天生非常适合 Seq2Seq Learning；由于 VAD 端点检测任务中，一帧属于语音还是静音与前后的帧均存在关系，故 BiLSTM 直观上也非常适合本任务，并且也已有利用 LSTM 做 VAD 检测的工作，得到了比以往模型更好的结果[1]。

在本实验中，我们基于双向 LSTM 构建了一个专用于端点检测的网络 VADnet。对于一个特定的音频特征矩阵，先随机初始化 LSTM 的 hidden state h_0 和 cell state c_0，然后将特征矩阵逐帧输入。LSTM 的每一个时间点接受输入 x_t 和上一时刻的 cell state c_{t-1}，经过门控机制的计算，保留一部分 cell memory，并用 x_t 计算出新的需要

[1] F. Eyben, F. Weninger, S. Squartini and B. Schuller, "Real-life voice activity detection with LSTM Recurrent Neural Networks and an application to Hollywood movies," 2013 IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 483-487, doi: 10.1109/ICASSP.2013.6637694.

```
VADnet(
  (lstm): LSTM(14, 32, bidirectional=True)
  (fc): Linear(in_features=64, out_features=1, bias=True)
  (sigmoid): Sigmoid()
)
```

图 2: *VADnet* 模型结构

加入 cell memory 的部分，产生这一时刻的 cell state c_t 供给下一时刻，并据此输出这一时刻的 hidden state h_t。经过一整个输入序列 $\mathbf{x}_N$ 之后，LSTM 输出一个序列 $\mathbf{o}_N = [h_1, h_2, ...h_N]$, 以及最终的 cell state c_N。这个输出序列 $\mathbf{o}_N$ 即可认为是对每一帧信息的一种编码表示。随后通过全连接层（Linear Layer），将每一个 h_t 变换为一个标量，通过 Sigmoid 激活函数得到 $[0, 1]$ 之间的概率输出。

综上，我们构建的 VADnet 由三个单元组成：双向 LSTM、全连接层和 Sigmoid 输出层。考虑到输入特征维度和计算量，我们设置隐层维度（hidden size）为 32，即每一个门控单元都由 32 个神经元组成，经过双向 LSTM 后会输出 32*2=64 维向量。为了模型的简单性，我们仅使用一层 LSTM。具体的模型结构如图2所示。

为了在训练过程中计算模型损失，进行梯度反传更新，我们令损失函数为二分类交叉熵损失函数（torch.nn.BCELoss），因为我们的 VADnet 实际上是一个二分类模型。BCELoss 的计算公式为

$$\mathcal{L}(\mathbf{p}_N, \mathbf{y}_N) = \frac{1}{N}\sum_{i=1}^{N}(y_i \log p_i + (1-y_i)\log(1-p_i)), \tag{1}$$

其中 $\mathbf{p}_N, \mathbf{y}_N$ 为预测序列和真实标签。随后，我们构建优化器为 Adam 优化器，学习率设置为 0.001。

构建模型之后，我们开始模型训练。我们将 train 文件夹下的 3600 条音频作为训练样本，dev 作为验证集，每一个 epoch 将 train 的 3600 条音频依次输入，计算 loss，进行梯度反传更新，然后计算 dev 集上的评测指标（AUC、EER）。本次实验没有使用 torch.utils.data.DataLoader，故为了避免数据顺序对模型参数的影响，在每个 epoch 前，人为 shuffle 打乱了数据列表。同时，每个 epoch 后，保存了模型参数。

我们只进行了两个 epoch 的训练，结果见下节所示。

3. 实验结果

实验发现，双向 LSTM 在 VAD 任务上得到了非常好的效果。图3展示了每个 epoch 之后在 dev 集上的 RoC 曲线，以及 AUC 和 EER 的值。可以发现，第一个 epoch 之后的 AUC 和 EER 已经非常接近 1.00，第二个 epoch 仅有微小的提升。这个结果说明 VADnet 并没有过拟合，因为模型没有在 dev 上做任何训练。

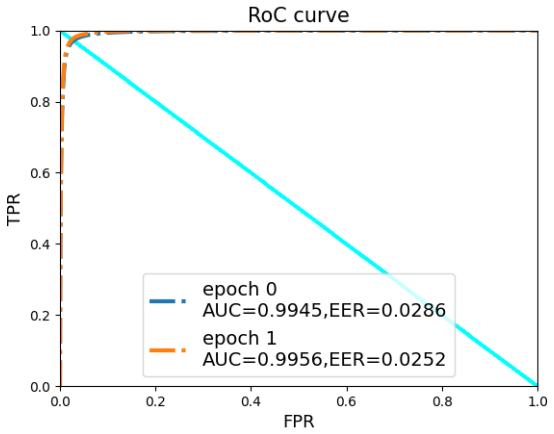

图 3: 两个 *Epoch* 后在 *dev* 集上的 *RoC* 曲线

为了更好地观察 VADnet 对于一些样本的预测情况，我们可视化 dev 上的几个样本的预测值和标签，如图 4、图5所示。

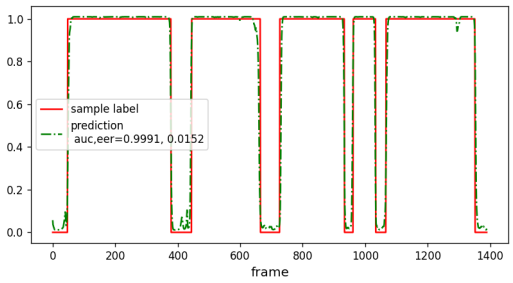

图 4: *dev/205-159056-0012.wav*

可以看出，VADnet 对于 VAD 标签的预测效果非常好，并且对于图5这样困难的任务，也能做到在端点上预测基本正确。更为神奇的是，这些

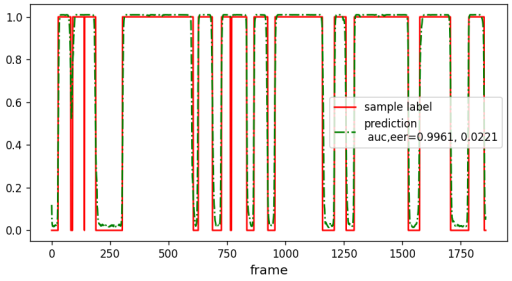

图 5: *dev/807-124223-0068.wav*

预测结果已经非常平滑，即并不存在以往逐帧预测那样容易出现很多不连续点的问题。因此我们在整个处理流程中也不需要做后处理，只需要在 test 预测的时候做一个简单的阈值分类即可。

进一步地，模型训练过程中 loss 随着 **batch** 变化的过程如表1。很明显，我们的 VADnet 其实不需要 3600 条那么多样本就可以达到很低的损失值。在第一个 epoch，VADnet 经过前 500 条音频之后已经能在没见过的音频上达到不错的预测效果。这揭示了 LSTM 对于 VAD 任务有完全足够的学习能力。

表 1: 训练过程 *loss* 变化趋势

Epoch	Batch	Avg Loss	Epoch	Batch	Avg Loss
0	50	0.5610	1	100	0.0560
0	100	0.3258	1	500	0.0528
0	150	0.1668	1	1000	0.0487
0	200	0.1227	1	1500	0.0546
0	500	0.0789	1	2000	0.0520
0	1500	0.0650	1	2500	0.0561
0	3550	0.0530	1	3550	0.0519

在 Evaluating 阶段，我们对预测结果进行阈值分类，计算帧级 Accuracy，并实现了童思博师兄提出的 VAD 评测指标[2]，计算了 SBA（Start Border Accuracy）、EBA（End Border Accuracy）、BP（Border Accuracy）以及最后合成的 VACC。我们在开发集上计算这些指标，在 SBA 和 EBA 的计算中取了区间长度 $L = 10$，结果如表2所示。

根据该论文中的结果，基于 DNN 实现的 VAD 算法当 VACC 达到 80% 时可以使 WER 降到 20% 以下，并且在工程应用中的 VAD 算法 VACC 达到 90% 以上已属优秀。本次实验使用的数据集较为干净，检测简单，因此能达到很高的 VACC。

表 2: 最终的评测指标

Metric	Value
AUC	0.9956
EER	0.0252
ACC	0.9820
SBA	0.9057
EBA	0.9075
BP	1.1380
VACC	0.9749

4. 思考和后记

本次实验通过 LSTM 的实践深刻体会到了其（包括其余种类的 RNN）在处理变长序列和建模前后文关系上的强大之处。Andrej Karpathy 的著名博客《The Unreasonable Effectiveness of Recurrent Neural Networks》[3]中即展现了 RNN 在诸多任务中的惊人表现，VAD 无疑是另一个有力佐证。当然 RNN 家族仍然存在一些不可避免的缺点，以 LSTM 为例，就包括自回归结构训练缓慢的问题（训练过程在超算上运行也需要 5 分钟左右，相较之以 GMM 模型为例的统计模型 VAD 算法略逊一筹）。本次实验将 batch size 设为 1，也是考虑到变长序列放在同一 batch 中需要经过其他较为复杂的处理（如 PyTorch 中的 pack_padded_sequence），这也在一定程度上降低了训练速度。所幸 LSTM 在这个任务上训练简单，只需少数样本就能做到很好的效果。

本实验也体验到 MFCC 对于信号特征表征的有效之处。其实之前尝试过直接用时域采样点作为每帧的输入 x_t，用 BiLSTM 训练，效果也能达到很高（AUC 在 98% 左右）。这提供了一个启示：完全端到端地做语音端点检测有多可行？或许需要更多的实验来探究。

[2]S. Tong, N. Chen, Y. Qian and K. Yu, "Evaluating vad for automatic speech recognition," 2014 12th International Conference on Signal Processing (ICSP), 2014, pp. 2308-2314, doi: 10.1109/ICOSP.2014.7015406.

[3]http://karpathy.github.io/2015/05/21/rnn-effectiveness/