Software Engineering and Applications 软件工程与应用, 2021, 10(4), 467-479
Published Online August 2021 in Hans. http://www.hanspub.org/journal/sea
https://doi.org/10.12677/sea.2021.104052

Hans汉斯

基于LSTM的端到端声纹识别算法实现

王　飞[1*]，徐颖捷[2]

[1]浙江理工大学信息学院，浙江　杭州
[2]浙江理工大学启新学院，浙江　杭州
Email: wf.09@qq.com, 1559175922@qq.com

收稿日期：2021年6月14日；录用日期：2021年7月21日；发布日期：2021年7月30日

摘　要

近年来，随着神经网络在语音识别领域应用中的快速发展，深度学习被应用到声纹识别领域，取得了很好的效果。本文先是介绍了声纹识别的基本理论，说明了语音信号预处理和特征识别的一般方法，而后又介绍了一种基于LSTM神经网络的端对端声纹识别算法，从理论上说明了这种算法的优越性。通过这种算法构建的说话人声纹识别模型，大大节省了模型训练的时间，训练效果较好。

关键词

声纹识别，端对端损失，LSTM神经网络

LSTM-Based End-to-End Voiceprint Recognition Algorithm Implementation

Fei Wang[1*], Yingjie Xu[2]

[1]Information College, Zhejiang Sci-Tech University, Hangzhou Zhejiang
[2]Qixin College, Zhejiang Sci-Tech University, Hangzhou Zhejiang
Email: wf.09@qq.com, 1559175922@qq.com

Received: Jun. 14th, 2021; accepted: Jul. 21st, 2021; published: Jul. 30th, 2021

Abstract

In recent years, with the rapid development of neural networks in the field of speech recognition applications, deep learning has been applied to the field of voiceprint recognition with good results. In this paper, we first introduce the basic theory of voiceprint recognition and illustrate the general methods of speech signal preprocessing and feature recognition, and then we introduce an

*第一作者。

文章引用: 王飞, 徐颖捷. 基于 LSTM 的端到端声纹识别算法实现[J]. 软件工程与应用, 2021, 10(4): 467-479.
DOI: 10.12677/sea.2021.104052

end-to-end voiceprint recognition algorithm based on LSTM Neural Network to illustrate the theoretical superiority of this algorithm. The speaker vocal pattern recognition model constructed by this algorithm greatly saves the time of model training and the training effect is better.

Keywords

Voiceprint Recognition, End-to-End Losses, LSTM Neural Network

1. 引言

1.1. 研究背景及意义

声纹识别是一种身份检测技术。由于声纹是携带信息的声波频谱，声纹同指纹一样，因此其具有独特的生物学特征，可用于身份识别。在文献[1]中，作者举例说明了说话人识别技术可以应用的领域，包括司法领域可以用来固定刑事侦查证据，确认犯罪嫌疑人；银行金融等安全领域可以用来核验身份，确定人员访问权限；电子通信及互联网领域可以用来登录 APP，减少了输入密码的麻烦。由此可见，说话人识别具有相当大的研究意义和使用价值，因此逐渐成为国内外研究学者关注的对象和研究热点。

1.2. 研究概况、水平和发展趋势

1945 年，Kersta 提出了“声纹(Voiceprint)”的概念。1969 年，J. E. Luck 在对语音特征分析的基础上，首次提出将倒谱技术应用到声纹识别技术中[2]，其实验结果较为理想。B. S. Atal 从中受到启发，他通过对声道进行分析建模提出了一系列参数，其中最著名的就是线性预测倒谱系数[3] (Linear Predictive Cepstrum Coefficients, LPCC)。同世纪的 80 年代 S. B. Davis 和 Hermansky 对人耳的听觉特性的分析和研究，并针对性地提出了 Mel 频谱的梅尔倒谱系数[4] (Mel Frequency Cepstral Coefficients, MFCC)。七十年代的矢量量化技术(Vector Quantization, VQ)在语音识别领域并取得了巨大的突破，随后 VQ 算法被应用于声纹识别领域[5]。为进一步提高识别结果，隐马尔科夫模型[6] (Hidden Markov Model, HMM)作为概率模型的代表被应用于声纹识别领域。随后，SVM、WCCN、NAP、LDA 等被用于声纹识别领域。2005 年，Kenny 提出联合因子分析[7] (Joint Factor Analysis, JFA)，在建模过程中将 GMM 的均值超矢量所包含的信息分解为两部分：说话人与说话人之间的差异(Speaker Variability, SV)，和相同说话人不同语音段之间的差异(Session Variability/Channels Variability, CV)。随后基于这个思想提出了一系列的基于向量的 i-vector 算法和基于信道补偿的 PLDA 算法。近几年，随着计算能力的快速提高，深度学习被越来越多的应用到声纹识别领域[8]，成绩斐然。以 ImageNet [9]为代表，深度学习神经网络在图像识别以及分类领域取得巨大成功，并诞生了一些经典的深度神经网络结构来解决通用问题，如 GoogLeNet，VGG，ResNet 等等。就语音识别领域来说，深度神经网络模型强大的拟合能力和泛化能力足以代替 GMM 模型，其模型建立和训练过程也变得足够简单。早期阶段有一些研究将 DNN 神经网络应用于说话人识别，用 DNN 代替 GMM 计算后验统计更改为高斯混合模型，延续了早期声纹识别的研究成果。

1.3. 本文主要研究内容以及章节安排

本文先是介绍了声纹识别的基本理论，说明了语音信号预处理和特征识别的一般方法，而后又介绍

了一种基于 LSTM 神经网络的端对端声纹识别算法，从理论上说明了这种算法的优越性。

本文的第一章是引言部分，主要介绍了说话人识别的研究背景及意义、研究概况，以及本文研究的主要内容。第二章介绍了声纹识别的基本理论，主要包括声纹识别的分类，语言信号的预处理以及语音信号的特征识别。第三章是论文的主要内容。主要介绍了循环神经网络的理论，端对端损失函数的原理，以及 LSTM 神经网络的结构。

2. 声纹识别基本理论

2.1. 声纹识别的分类

作为模式识别的一类，声纹识别的主要任务是通过待测试语音来判断对应说话人身份。声纹识别可以分为两类。若已知待测说话人的范围，需要通过随机或者特定的语音段来判断是否属于某个说话人，这属于声纹确认技术。这种问题是 1 对 1 的身份判别问题。若待测说话人的身份范围没有确定，需要通过随机或者特定的语音段来确定说话人的身份，这属于声纹辨认技术。这种问题是 1 对 N 的身份辨别问题。

声纹识别问题又可以分为与文本相关的声纹识别和文本无关的声纹识别。对于文本相关问题，待测试语音段的内容需要和系统中预先登记的内容相同。对于文本无关问题，待测试语音段的内容可以与系统中预先登记的内容不同，待测试说话人可以只说几个字来进行身份认证。本文涉及的是 1 对 1 的、文本无关的身份判别问题。

2.2. 语音信号的预处理和特征识别

2.2.1. 语音信号的预处理

为了消除一些影响音频信号质量的因素，在进行声纹识别之前，一般要对语音信号进行预处理。语音信号的预处理一般包括预加重、分帧和加窗等。一般情况下，经过语音预处理后得到的信号会更加均匀、流畅，频谱更清晰。语音信号经过预处理，可以增强语音处理的效果。

1) 预加重

由于语音信号的高频部分在传输过程中会衰减，这种衰减严重影响声纹识别的效果，所以必须在传输线路的起始端对信号的高频部分进行增强。这种技术被称为预加重技术预加重技术可以有效地提高输出信噪比，因为其对噪声没有影响。一般说来，将信号通过一阶有限冲激响应高通数字滤波器就可以实现。数字滤波器的系统函数为：

$$H(z)=1-az^{-1} \tag{1}$$

2) 分帧加窗

为了对语音信号进行频域分析，傅里叶变换是一种常见的工具。傅里叶变换的存在条件是信号经历的随机过程是平稳的，但是从宏观层面看，要求信号平稳的这个条件过于苛刻。在微观层面看，当信号被限定在一个较短时间内，就可以视为平稳信号。把信号的连续若干点设为一帧，这样的操作被称为分帧。分帧后的信号还不能马上进行傅里叶变换。由于分帧后的语音信号不光滑，信号的分辨率较差，因此在做傅里叶变换之前，还要先对信号乘以一个窗函数。这样的操作被称为加窗。加窗的目的是为了让分帧后的信号的两端无限接近于 0。此项操作可以提高变换结果的分辨率。但加窗的会削弱信号两端的部分，所以在分帧时可以考虑相互重叠，重叠的部分一般是帧长的一半，这部分被称为帧移。

常见的窗函数有两种，分别是矩形窗和汉明窗。实验中采取的窗函数是汉宁窗，如图 1 所示。

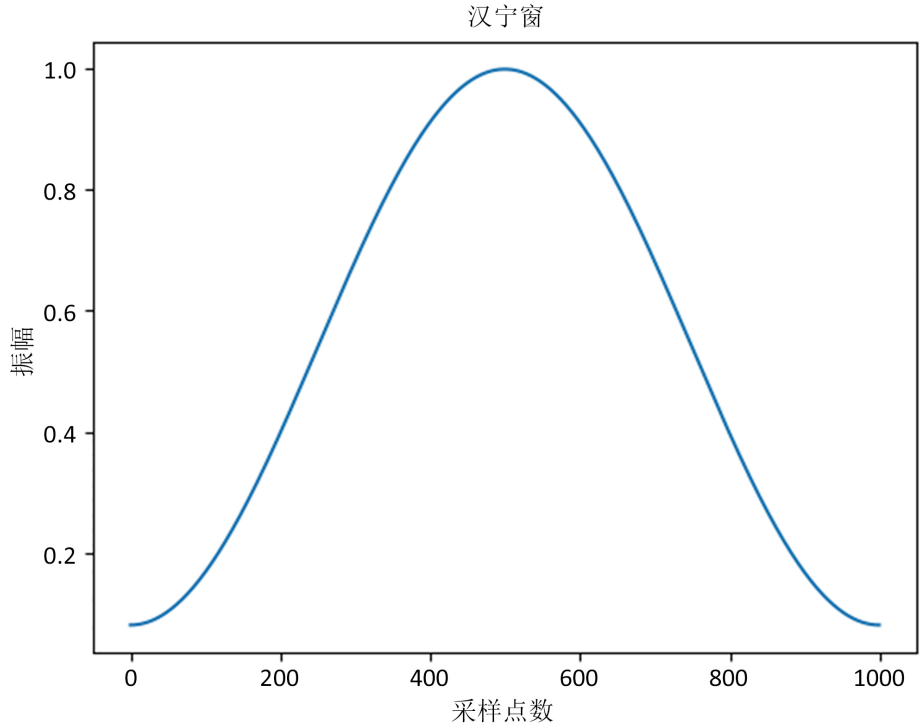


Figure 1. Hanning window function
图 1. 汉宁窗函数

3) 语音活动检测

由于语音信号中即存在活动的部分又存在非活动的部分，为了区分这两种语音部分，语音活动检测技术应运而生。语音活动检测技术可以检测出原始语音信号中活动的语音，即存在人类语言的大部分，减少计算量。语音活动检测技术通常是与语音无关的。传统的语音活动检测的方法主要是提取语音信号的时域特征，例如短时能量、过 0 率、相关性和基音等[10]。在信号的信噪比较高的情况下，语音活动检测的效果较好。

实验采取的是根据短时能量的大小来分辨语音信号和非语音信号。实验中利用 Python 语言中 librosa 工具包中的 librosa.effects.split 函数，可以将信号分成非静音间隔，方便后续处理。

2.2.2. 语音特征提取

为了将语音信号转换为计算机能够处理的特征向量，需要提取语音信号的参数特征，这样的操作被称为语音特征提取。常见的语音特征提取方法有以下几种。

1) 线性预测分析(Linear predictive coding, LPC)

线性预测分析最早由 Weiner 于 1967 年提出，后被应用于许多研究领域[11]。

线性预测分析的思路就是通过线性组合过去的语音样本，使得线性预测样本的误差和实际语音样本的误差平方和最小，进而可以确定线性预测系数。也就是说，通过线性预测分析，可以用模型或信号的输出描述语音信号。系统的传输函数可以表示为：

$$H(z)=\frac{G}{1-\sum_{i=1}^{r}a_i z^{-i}} \tag{2}$$

这种系统称为全极点系统。其中分母多项式的系数称为线性预测系数。这样的信号模型在忽略一定精度情况下可以描述信号。线性预测分析就是在某个准则下，根据已知的语音信号对参数与进行参数估

计。线性预测分析首先要解决的问题是如何通过语音信号确定参数集$\{a_i\}$，使预测误差最小。一般采用最小均方误差准则进行参数估计。由于全极点模型易于计算，对其进行参数估计求解线性方程组，较易实现，因此线性预测模型采用全极点模型较为合理。

若该系统函数的分母多项式能够快速收敛，则仅仅需要前几项。因此，在实际应用中，可以用全极点模型近似表示零极点模型。

根据上述模型化思想，可对语音信号建立模型，将其中的声门激励、声道以及全部谱效应简化为一个时变数字滤波器来等效。其系统函数为：

$$H(z)=\frac{S(z)}{U(z)}=\frac{G}{1-\sum_{i=1}^{r}a_i z^{-i}} \tag{3}$$

2) 线性预测倒谱分析(Linear prediction cepstrumcoding, LPCC)

线性倒谱系数是 LPC 参数在倒数谱中的表示，能够很好地反应人的声道特征[10]。下面是由线性预测系数求解线性预测倒谱系数的递推公式，其中a_k表示 LPC 系数。

$$c(n)=\begin{cases}0 & n<0\\ \ln(G) & n=0\\ a_n+\sum_{k=1}^{n-1}\left(\frac{k}{n}\right)C(k)a_{n-k} & 0<n\le p\\ \sum_{k=n-p}^{n-1}\left(\frac{k}{n}\right)C(k)a_{n-k} & n>p\end{cases} \tag{4}$$

3) 梅尔频率倒谱分析(Mel frequency cepstrumcoding, MFCC)

为了更好的处理语音信号，往往可以考虑将语音信号的时频谱转换为梅尔谱。转换的方法是把信号通过一组梅尔标度滤波器。声学理论可知，人耳对语音信号的感知不是线性关系。如果我们将声谱图转换为梅尔谱，这种非线性的感知关系将转换为线性关系。梅尔频率与普通频率的关系如下：

$$Mel(f)=2565\times\log_{10}\left(1+\frac{f}{700}\right) \tag{5}$$

MFCC 特征参数的提取过程如下：

1、首先，对预处理后的信号进行快速傅里叶变换，得到各帧的频谱，平方以后可以得到语音信号的能量谱。

2、将得到的能量谱通过一组梅尔滤波器，其中滤波器的频率响应为：

$$H_m(k)=\begin{cases}0 & k<f(m-1)\text{ or }k>f(m-1)\\ \frac{2(k-f(m-1))}{(f(m+1)-f(m-1))(f(m)-f(m-1))} & f(m-1)\le k\le f(m)\\ \frac{2(f(m+1)-k)}{(f(m+1)-f(m-1))(f(m)-f(m-1))} & f(m)\le k\le f(m+1)\end{cases} \tag{6}$$

其中$f(m)$为中心频率，$m=1,2,3,\cdots,M$. M 通常取 22~26。

3、计算每个滤波器输出的对数值：

$$S(m)=\ln\left(\sum_{k=0}^{N-1}\left|X_a(k)\right|^2 H_m(k)\right),0\le m\le M \tag{7}$$

由此可以得到 L 阶 MFCC 系数，一般 L 取 12~16 之间。

MFCC 特征参数提取过程如图 2 所示。

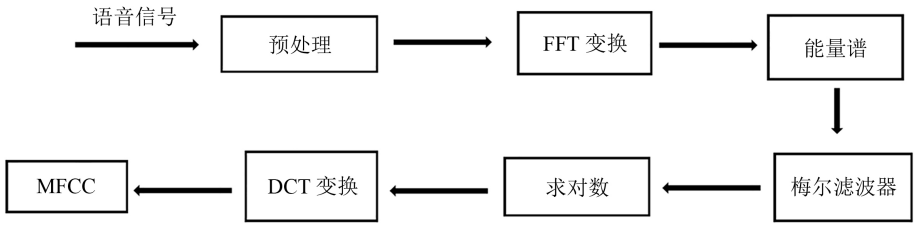


Figure 2. MFCC feature parameter extraction process diagram
图 2. MFCC 特征参数提取过程图

3. 基于 LSTM 神经网络的说话人识别研究

3.1. 长短时记忆网络 LSTM

近年来，循环神经网络越多地被应用到语音信号处理领域。随着应用地不断深入，循环神经网络的弊端开始显现。由于长时间学习而导致的梯度弥散或者梯度爆炸问题被越来越多的学者所揭示。为了解决循环神经网络出现的问题，有学者提出了长短时记忆网络[12] (Long short-term memory, LSTM)。在循环神经网络的基础上，长短时记忆网络改进了循环神经网络权值的计算规则。与普通的循环神经网络相比，LSTM 能够学习长期依赖信息。

标准的循环神经网络具有一个单一的、完全相同的神经网络层。常见的是单 tanh 层，如图 3 所示。

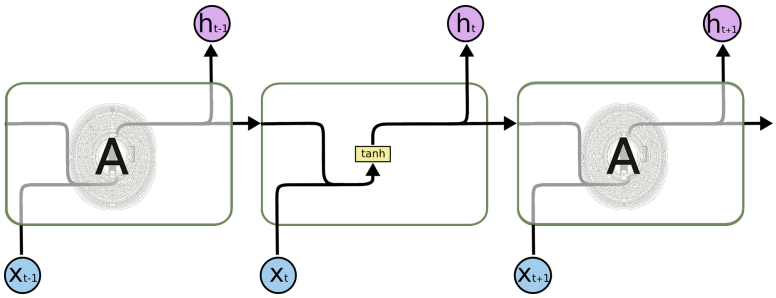


Figure 3. A simple recurrent neural network pattern diagram
图 3. 一种简单的循环神经网络模式图

而对于 LSTM 来说，情况则大不相同。这里不再是单一的 tanh 层，而是有四个不同的结构，分别是 1 个 tanh 层和 3 个 sigmoid 层，如图 4 所示。

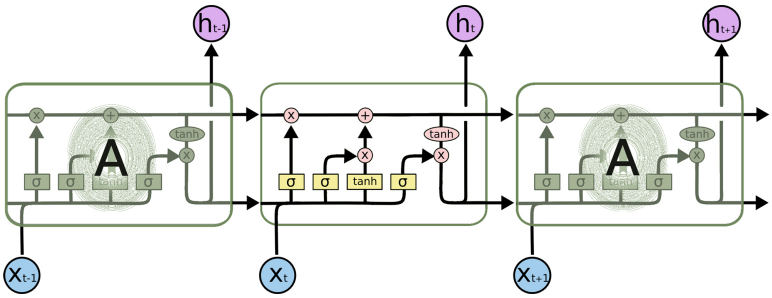


Figure 4. A simple long- and short-term memory network model diagram
图 4. 一种简单的长短时记忆网络模式图

通过引入一种控制结构，神经网络可以学会如何控制信息去除或者增加到细胞(cell)中。也就是说，神经网络可以让信息有选择地通过。这种结构包含了一个 sigmoid 神经网络层和一个乘法操作，如图 5 所示。

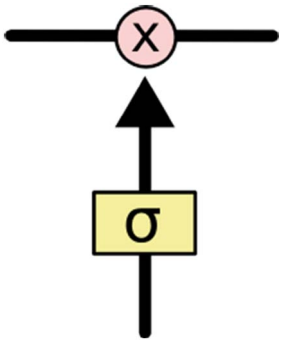


Figure 5. Control structure in LSTM
图 5. LSTM 中的控制结构

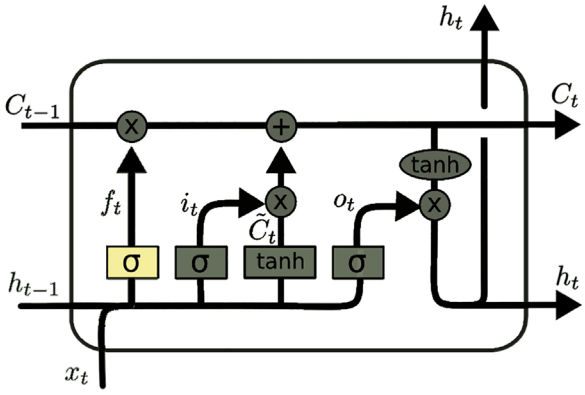
Sigmoid 神经网络层会输出 0 到 1 之间的数字，0 意味着禁止信息通过，1 意味着允许所有信息通过。一个 LSTM 神经网络中有三个类似的控制结构，来控制神经网络中细胞的状态(state)。

对图 4 所示的神经网络计算过程如下：

第一步，从细胞状态中丢弃旧信息。此行为通过一个被称作是忘记门层的控制结构来完成。忘记门层会读取 h_{t-1} 和 x_t，计算公式为

$$f_t = \sigma\left(W_f \times \left[h_{t-1}, x_t\right] + b_f\right) \tag{8}$$

如图 6 所示，可以得到一个在 0 到 1 之间的数值，此时的神经网络状态记为 C_{t-1}。



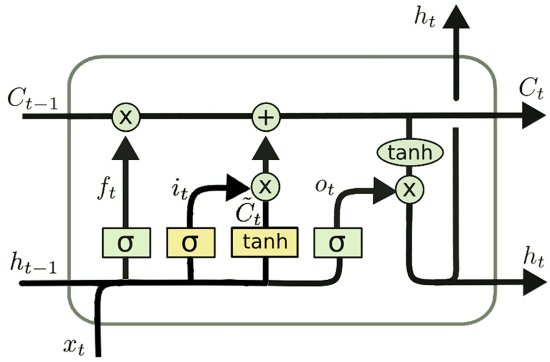
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

Figure 6. The first step of LSTM neural network
图 6. LSTM 神经网络第一步运算

第二步，存新信息到细胞状态中。控制结构的一部分是 sigmoid 层，该层决定什么值需要更新。另一部分的结构是 tanh 层。当信息通过 tanh 层时会得到一个新的向量，记为 $\tilde{C}_t$，被记入到细胞状态中。细胞会根据这两个值进行更新，如图 7 所示。这两部分的计算公式为：

$$i_t = \sigma\left(W_f \times \left[h_{t-1}, x_t\right] + b_f\right) \tag{9}$$

$$\tilde{C}_t = \tanh\left(W_c \times \left[h_{t-1}, x_t\right] + b_c\right) \tag{10}$$

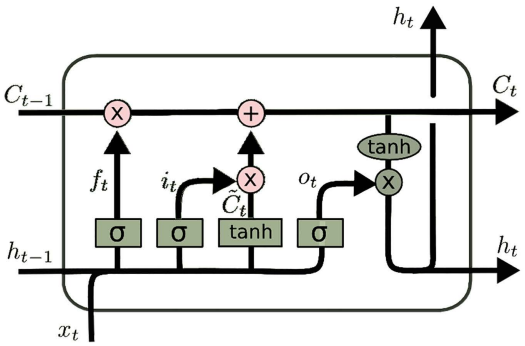


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \ + \ b_C)$$

Figure 7. The secondstep of LSTM neural network
图 7. LSTM 神经网络第二步运算

第三步，更新细胞状态。将旧的细胞状态与 f_t 相乘，丢弃掉决定要丢弃的信息，再加上 $i_t \times \tilde{C}_t$，这就是新的细胞状态，如图 8 所示。计算公式为：

$$C_t = f_t \times C_{t-1} + i_t \times \tilde{C}_t \tag{11}$$



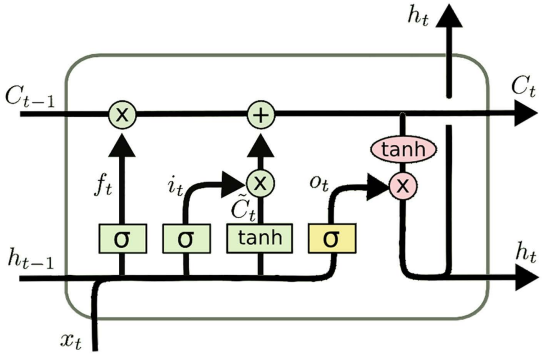
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

Figure 8. The third step of LSTM neural network
图 8. LSTM 神经网络第三步运算

第四步，计算细胞输出。输出来自于新的细胞状态，如图 9 所示。计算公式为：

$$o_t = \sigma\left(W_o \times \left[h_{t-1}, x_t\right] + b_o\right) \tag{12}$$

$$h_t = o_t \times \tanh\left(C_t\right) \tag{13}$$



$$o_t = \sigma\left(W_o \ [h_{t-1}, x_t] \ + \ b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

Figure 9. The fourth step of LSTM neural network
图 9. LSTM 神经网络第四步运算

至此，一层简单的 LSTM 神经网络创建完成。实验中通过 Python 语言的 Pytorch 工具包中的 torch.nn.LSTM()函数即可快速构建 LSTM 神经网络，如图 10 所示。

```
def __init__(self):
    super(SpeechEmbedder, self).__init__()
    self.LSTM_stack = nn.LSTM(hp.data.nmels, hp.model.hidden, num_layers=hp.model.num_layer, batch_first=True)
    for name, param in self.LSTM_stack.named_parameters():
      if 'bias' in name:
         nn.init.constant_(param, 0.0)
      elif 'weight' in name:
         nn.init.xavier_normal_(param)
    self.projection = nn.Linear(hp.model.hidden, hp.model.proj)
```

Figure 10. Use the nn.LSTM function to quickly create an LSTM neural network
图 10. 使用 nn.LSTM 函数快速创建 LSTM 神经网络

3.2. 基于三元组的端对端损失函数的声纹识别系统

参考文献[13]中介绍了一种基于三元组的端到端损失函数(the end-to-end loss function based on triples, TE2E)声纹识别系统。

作者将评价话语 $X_{j\sim}$ 和 M 个注册话语 $X_{km}\left(m=1,2,\cdots,M\right)$ 作为一个元组送入 LSTM 神经网络。其中，X 是从定长段语音信号中提取的梅尔频谱系数，j 和 k 代表话语的说话者。两者相等的关系不确定。若两者相等，则说明来自同一个说话者，认为该元组是正元组(positive)；反之，认为该元组是负元组(negative)。

对于每个输入的元组，作者计算了 LSTM 神经网络的 L2 正则(the L2 normalized)：$\left\{e_{j\sim},\left(e_{k1},\cdots,e_{kM}\right)\right\}$。这里的每一个 e 是固定维度的嵌入矢量(d-vector)。它的维度由 LSTM 神经网络的投影层大小决定。元组的中心表示从 M 个话语构建的声纹(voiceprint)，计算过程如下：

$$c_k=E_m\left[e_{km}\right]=\frac{1}{M}\sum_{m=1}^{M}e_{km} \tag{14}$$

使用余弦相似度函数定义两者之间的相似度：

$$s=w\cdot\cos\left(e_{j\sim},c_k\right)+b \tag{15}$$

其中 w 和 b 分别是神经网络可以学习的权值和偏置。最终损失函数被定义为：

$$L_T\left(e_{j\sim},c_k\right)=\delta\left(j,k\right)\sigma\left(s\right)+\left(1-\delta\left(j,k\right)\right)\left(1-\sigma\left(s\right)\right) \tag{16}$$

其中 $\sigma\left(x\right)=1/\left(1+\mathrm{e}^{-x}\right)$ 是标准的 sigmoid 函数。若 $j=k$，则 $\delta\left(j,k\right)=1$，反之，$\delta\left(j,k\right)=0$。当 $j=k$ 时，TE2E 损失函数会使相似度 s 越来越大；当 $j\neq k$ 时，TE2E 损失函数会使相似度越来越小。这两种元组的更新方式和在 FaceNet [14]中使用的正负元组非常类似。

3.3. 改进后的端对端损失函数的声纹识别系统

在本章中，介绍了 TE2E 损失函数的一种改进后的形式(a generalization of our TE2E architecture)。我们称其为 GE2E 损失函数。这种改进后的算法以一种更加有效的方式生成嵌入矢量，这方式显著提高了与文本无关的说话人验证(TI-SV)的性能和训练速度。

此次提出的模型是基于批量处理的语音信号，此举大大节省了运算时间。模型训练时的每个批次(batch)包含 N 个说话者，平均每个说话者包含 M 个话语，如图 11 所示。

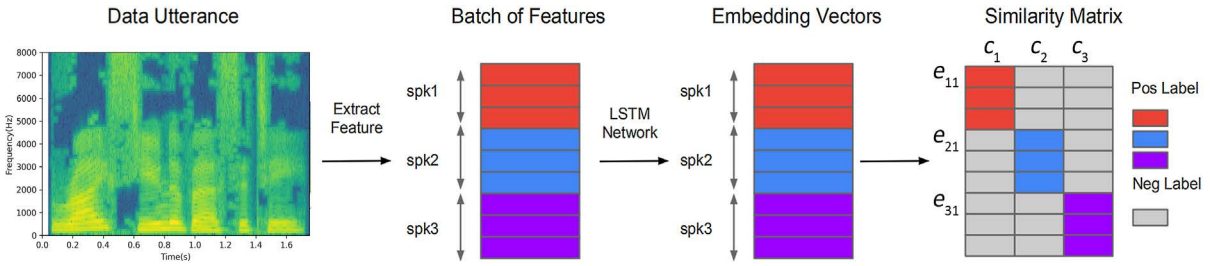


Figure 11. Overview of Voiceprint Recognition System
图 11. 声纹识别系统概述

3.3.1. 广义的端到端模型

模型采取大小的话语来构造一个批次(batch)。每个说话者都有 M 个不同的话语，共计有 N 个不同的说话者。这些构成了 $N\times M$ 的矩阵。记特征向量为 X_{ji}，其中 j 和 i 的范围分别是$1\le j\le N$ 和$1\le i\le M$。每一个特征向量说明了特征来自第 j 个说话者的第 i 个的话语。

将特征向量送入 LSTM 神经网络。LSTM 神经网络的最后一层连接了一个线性层，用来降低特征向量的维度。将公式 $f\left(X_{ji};W\right)$ 定义为神经网络的输出，其中神经网络中所有可以学习的参数被记为 W，包括 LSTM 层和线性层的参数。由网络输出的 L2 正则形式可以得到嵌入矢量(d-vector)，嵌入矢量的计算公式为：

$$e_{ji}=\frac{f\left(X_{ji};W\right)}{\left\|f\left(X_{ji};W\right)\right\|_2} \tag{17}$$

这里的 e_{ji} 代表着第 j 个说话者的第 i 个话语的嵌入矢量。定于所有嵌入矢量的中心为该说话者的声纹。记 c_j 为第 j 个说话者的声纹，计算公式为：

$$c_j=c_k=E_m\left[e_{km}\right]=\frac{1}{M}\sum_{m=1}^{M}e_{km} \tag{18}$$

定义每个嵌入矢量 e_{ji} 和嵌入矢量的中心 c_k 之间的余弦相似度为相似度矩阵 $S_{ji,k}$，计算公式为：

$$S_{ji,k}=w\cdot\cos\left(e_{ji},c_k\right)+b \tag{19}$$

其中 w 和 b 分别是神经网络可以学习的权值和偏置。为了余弦相似度越大时，相似度矩阵也越大，我们限定 $w>0$。与改进之前的损失函数相比，TE2E 和 GE2E 损失函数的主要区别如下：

1、由公式 17 可知，TE2E 的相似度是一个标量值。它定义了嵌入矢量和单个元组中心的相似度。

2、由公式 19 可知，GE2E 的相似度是矩阵形式。它定义了嵌入矢量和所有元组中心之间的相似度。

图 1 展示了整个过程，包括来自不同说话人的特征、嵌入矢量以及相似度评分，并用不同的颜色表示。

训练过程中，我们希望每个话语的嵌入矢量与该说话者的所有话语的中心尽可能地靠近，同时尽可能地远离其他说话者的中心。图 12 的相似性矩阵说明了这个问题。我们希望主对角线上的相似值越大越好，而其他部分的相似值越小越好。图 13 以另一种形式说明了这个问题。我们希望蓝色嵌入矢量尽可能的靠近它自己说话者嵌入矢量的中心，即蓝色三角形；同时尽可能的远离其他说话者的中心，即紫色和红色三角形。尤其是需要远离距离它最近的红色三角形。如图 12 所示，给定一个嵌入矢量 e_{ji}，所有的说话者嵌入矢量的中心即说话者的声纹 c_k，以及所有的相似度矩阵 $S_{ji,k}$，有两种损失函数可以实现这种要求。

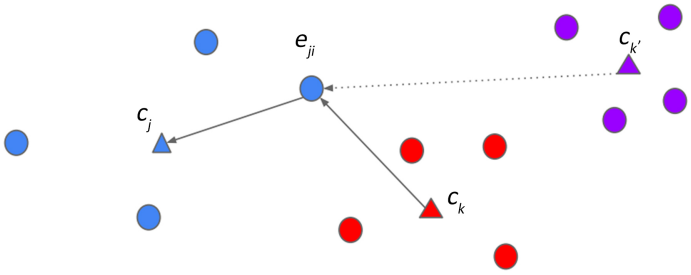


Figure 12. Schematic diagram of embedding vector and voiceprint of true and false speakers
图 12. 嵌入矢量与真假说话者声纹的示意图

1) Softmax.在相似度矩阵上设置一个交叉熵损失函数，其中 $k=1,2,\cdots,N$ 。若 $j=k$ ，则输出等于 1，否则输出等于 0。因此，每个嵌入矢量 e_{ji} 上的损失可定义为：

$$L\left(e_{ji}\right)=-S_{ji,i}+\log\sum_{k=1}^{N}\exp\left(S_{ji,k}\right) \tag{20}$$

这个损失函数使得每个嵌入矢量距离说话人中心越来越近，并远离其他说话人的中心。

2) Contrast.在正元组和距离最远的负元组之间定义一个对比度损失函数：

$$L\left(e_{ji}\right)=1-\sigma\left(S_{ji,j}\right)+\max_{\substack{1\le k\le N\\ k\ne j}}\left(S_{ji,k}\right) \tag{21}$$

其中 $\sigma(x)=1/\left(1+\mathrm{e}^{-x}\right)$ 是 sigmoid 函数。对于每一个话语，有两部分组成了损失。这两部分分别是：

1、一个正项，它与嵌入矢量和它的真说话人声纹之间的正匹配呈现相关关系。

2、一个权重较大的负项，它与嵌入矢量和在所有假说话者中相似性最高的说话人声纹之间的负匹配呈现相关关系。

在图 13 中，正项对应于使嵌入矢量靠近真说话人的声纹。因为与相比更靠近嵌入矢量，所以负项对应于使嵌入矢量远离所有假说话者中相似性最高的假说话人的声纹。因此，对比度损失函数使得嵌入矢量和假说话人的声纹之间的相似度得到重点关注。

在此次实验中，我们发现使用 softmax 损失函数在与文本无关的说话人识别问题中表现更好。

此外，我们还观察到，在计算真正说话者的声纹时，一种简单的方案是去掉 e_{ji} 。这样会使得模型更容易收敛。因此，我们在计算负相似性 $\left(k\ne j\right)$ 时仍然使用公式 16；当 $k=j$ 时，我们使用公式 22：

$$c_j^{(-i)}=\frac{1}{M-1}\sum_{\substack{m=1\\ m\ne i}}^{M}e_{jm} \tag{22}$$

$$S_{ji,k}=\begin{cases}w\cdot\cos\left(e_{ji},c_j^{(-i)}\right)+b & k=j\\ w\cdot\cos\left(e_{ji},c_k\right)+b & \text{otherwise}\end{cases} \tag{23}$$

结合公式 17，20，21 和 22，最终的 GE2E 损失 L_G 是相似矩阵上所有损失的总和：

$$L_G\left(x;w\right)=L_G\left(S\right)=\sum_{j,i}L\left(e_{ji}\right) \tag{24}$$

其中 $1\le j\le N,1\le i\le M$ 。

3.3.2. GE2E 和 TE2E 的比较

考虑 GE2E 的单个批次的损失更新：每个批次有 N 个说话人，每个说话人有 M 个话语。每一步更新都

会是所有的大小为 $N\times M$ 的嵌入向量尽可能地靠近它们自己说话者的中心，并使其尽可能地远离其他说话者的中心。这反映了 TE2E 损失函数[13]中每个 X_{ji} 的所有可能元组的情况。假设我们随机选择 P 个话语：

1、正元组：记为 $\left\{X_{ji},\left(X_{j,i_1},\cdots,X_{j,i_P}\right)\right\}$，其中 $1\le i_p\le M$，$p=1,\cdots,P$。一共有 $\binom{M}{P}$ 个这样的元组。

2、负元组：记为 $\left\{X_{ji},\left(X_{k,i_1},\cdots,X_{k,i_P}\right)\right\}$，其中 $k\ne j$，$1\le i_p\le M$，$p=1,\cdots,P$。对于每一个 X_{ji}，我们与其他所有的 $N-1$ 个中心进行比较。其中，每一组比较都包含 $\binom{M}{P}$ 个这样的元组。

3、一个正元组对应一个负元组，所以元组的总数是正元组和负元组的最大数目的 2 倍。因此 TE2E 损失的元组总数：

$$2\times\max\left(\binom{M}{P},(N-1)\binom{M}{P}\right)\ge 2(N-1) \tag{25}$$

当 $P=M$ 时，公式 25 出现最小值。因此 GE2E 中每次更新 X_{ji} 等价于 TE2E 中至少更新 $2(N-1)$ 次。上述分析说明了为什么 GE2E 模型要优于 TE2E 模型。

3.4. 实验结果及分析

本实验使用的数据集为 TIMIT 语音数据集，它是由美国德州仪器公司、麻省理工学院和 SRI 国际公司构建的语音库。其中全部语音信号的采样频率都是 16 kHz，一共包含 6300 个句子。实验采用了完整的 TIMIT 数据集。

实验中特征提取过程与[15]一致。首先将音频信号转换为帧长为 25，帧移为 10 的帧。然后提取 40 维度的梅尔频率倒谱系数作为每个帧的特征。对每个话语进行语音活动检测，然后选择每个话语的前 180 帧和后 180 帧，提取 MPCC 特征，然后送入神经网络。

使用带有投影层(projection) [16]的三层 LSTM 神经网络，嵌入矢量的维度和投影层的维度相同。对于此次文本无关的声纹识别，隐藏层大小被设置为 768，投影层大小被设置为 256。训练时每批次包含 $N=4$ 个说话者，$M=5$ 个话语。训练方法采用随机梯度下降法，训练的学习率被设置为 0.01。同时，将梯度的 L2 范数裁剪[17]为 3。对于权值 (W,b)，我们还观察到一个较小的初值是 $(W,b)=(10,-5)$。较小的梯度有助于平滑收敛。如图 13 所示，可以看到，训练一共迭代到 133,929 次，损失下降到一个较低水平。

```
Mon Apr 13 15:14:21 2020  Epoch:950[120/141],Iteration:133929  Loss:0.0320  TLoss:0.4443

Done, trained model saved at ./speech_id_checkpoint/final_epoch_950_batch_id_141.model
```

Figure 13. Experiment log results
图 13. 训练日志结果

测试时每批次包含 $N=4$ 个说话者，$M=6$ 个话语。如图 14 所示，从测试日志中可以看到等错误率(ERR)始终维持在一个较低水平，模型训练效果较好。

```
EER : 0.08 (thres:0.51, FAR:0.08, FRR:0.08)

EER : 0.00 (thres:0.50, FAR:0.00, FRR:0.00)

EER across 10 epochs: 0.0387
```

Figure 14. Test log results
图 14. 测试日志结果

4. 结束语

在本文中，我们提出了一种改进后的端对端损失函数，这种损失函数与前人提出的损失函数相比，可更有效地训练声纹识别模型。我们从理论上验证了这种损失函数的优越性，并用 Python 语言实现了它。通过使用这种改进后的端对端损失函数，我们得到了更精确的声纹识别模型。

参考文献

[1] 樊云云. 面向说话人识别的深度学习方法研究[D]: [硕士学位论文]. 南昌: 南昌航空大学, 2019.

[2] Luck, J.E. (1969) Automatic Speaker Verification Using Cepstral Measurements. *Journal of the Acoustical Society of America*, **46**, 1026-1032. https://doi.org/10.1121/1.1911795

[3] Atal, B.S. (1976) Automatic Recognition of Speakers from Their Voices. *Proceedings of the IEEE*, **64**, 460-475. https://doi.org/10.1109/PROC.1976.10155

[4] Davis, S. and Mermelstein, P. (1980) Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences. *IEEE Transactions on Signal Processing*, **28**, 357-366. https://doi.org/10.1109/TASSP.1980.1163420

[5] Sakoe, H. and Chiba, S. (1978) Dynamic Programming Algorithm Optimization for Spoken Word Recognition. *IEEE Transactions on Acoustics, Speech and Signal Processing*, **26**, 43-49. https://doi.org/10.1109/TASSP.1978.1163055

[6] Matsui, T. and Furui, S. (1994) Comparison of Text-Independent Speaker Recognition Methods Using VQ-Distortion and Discrete/Continuous HMM's. *IEEE Transactions on Speech and Audio Processing*, **2**, 456-459. https://doi.org/10.1109/89.294363

[7] Kenny, P. (2005) Joint Factor Analysis of Speaker and Session Variability: Theory and Algorithms.

[8] Lei, Y., Scheffer, N., Ferrer, L. and McLaren, M. (2014) A Novel Scheme for Speaker Recognition Using a Phonetically-Aware Deep Neural Network. 2014 *IEEE International Conference on Acoustics, Speech and Signal Processing*, Florence, 4-9 May 2014, 1695-1699. https://doi.org/10.1109/ICASSP.2014.6853887

[9] Deng, J., Dong, W., Socher, R., *et al*. (2009) ImageNet: A Large-Scale Hierarchical Image Database. *IEEE Conference on Computer Vision and Pattern Recognition*, Miami, 20-25 June 2009, 248-255. https://doi.org/10.1109/CVPR.2009.5206848

[10] 刘华平, 李昕, 徐柏龄, 姜宁. 语音信号端点检测方法综述及展望[J]. 计算机应用研究, 2008(8): 2278-2283.

[11] 胡航. 现代语音信号处理[M]. 北京: 电子工业出版社, 2014: 74.

[12] Hochreiter, S. and Schmidhuber, J. (1997) Long Short-Term Memory. *Neural Computation*, **9**, 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[13] Heigold, G., Moreno, I., Bengio, S. and Shazeer, N. (2016) End-to-End Text-Dependent Speaker Verification. 2016 *IEEE International Conference on in Acoustics, Speech and Signal Processing*, Shanghai, 20-25 March 2016, 5115-5119. https://doi.org/10.1109/ICASSP.2016.7472652

[14] Schroff, F., Kalenichenko, D. and Philbin, J. (2015) Facenet: A Unified Embedding for Face Recognition and Clustering. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, Boston, 7-12 June 2015, 815-823. https://doi.org/10.1109/CVPR.2015.7298682

[15] Prabhavalkar, R., Alvarez, R., Parada, C., Nakkiran, P. and Sainath, T.N. (2015) Automatic Gain Control and Multi-Style Training for Robust Small-Footprint Keyword Spotting with Deep Neural Networks. 2015 *IEEE International Conference on Acoustics, Speech and Signal Processing*, South Brisbane, 19-24 April 2015, 4704-4708. https://doi.org/10.1109/ICASSP.2015.7178863

[16] Sak, H., Senior, A. and Beaufays, F. (2014) Long Short-Term Memory Based Recurrent Neural Network Architectures for Large Vocabulary Speech Recognition. https://arxiv.org/abs/1402.1128

[17] Pascanu, R., Mikolov, T. and Bengio, Y. (2012) On the Difficulty of Training Recurrent Neural Networks. https://arxiv.org/abs/1211.5063