语音识别中的深度学习方法

(申请清华大学工学硕士学位论文)

培 养 单 位：计 算 机 科 学 与 技 术 系
学　　科：计 算 机 科 学 与 技 术
研　究　生：刘　超
指 导 教 师：郑　方　研　究　员

二〇一五年六月

Deep Learning Methods in Automatic Speech Recognition

Thesis Submitted to

Tsinghua University

in partial fulfillment of the requirement

for the degree of

Master of Science

in

Computer Science and Technology

by

Liu Chao

Thesis Supervisor : Professor Zheng Fang

June, 2015

关于学位论文使用授权的说明

本人完全了解清华大学有关保留、使用学位论文的规定，即：

清华大学拥有在著作权法规定范围内学位论文的使用权，其中包括：（1）已获学位的研究生必须按学校规定提交学位论文，学校可以采用影印、缩印或其他复制手段保存研究生上交的学位论文；（2）为教学和科研目的，学校可以将公开的学位论文作为资料在图书馆、资料室等场所供校内师生阅读，或在校园网上供校内师生浏览部分内容。

本人保证遵守上述规定。

（保密的论文在解密后应遵守此规定）

作者签名：_____　　导师签名：_____

日　　期：_____　　日　　期：_____

摘 要

语音识别是一项重要的人机交互技术，广泛地应用于各种各样的日常生活场景之中。而近几年深度神经网络在语音识别领域的应用大幅度地提高了已有语音识别系统的准确率，使得语音识别技术变得更加可用。尽管如此，环境噪声等因素对语音识别系统性能的影响仍然很大，阻碍了语音识别技术在实际环境中更大范围的应用。因此，如何更好的使用机器学习方法提高系统性能是一个重要的研究课题。

本文主要研究如何更好地在大规模连续语音识别系统中应用深度学习方法，提高系统性能，加快运算速度。研究和工作主要由三个方面构成：

首先，我们提出了一种对 HMM 混合模型中的深度神经网络进行网络权重裁剪的方法。网络权重裁剪可以减小模型的体积，增强模型的可扩展性。该方法以最小化目标函数的变化为目标，利用对角化 Hessian 矩阵假设进行网络的裁剪过程。对比实验表明该裁剪方法相比直接按权重大小的裁剪可以获得更好的性能。

接下来，我们提出了一种利用深度前馈神经网络迁移训练时间递归神经网络的方法。时间递归神经网络可以记忆长时的上下文信息，非常适合对语音这种序列化信号的建模。但是，时间递归神经网络在大规模连续语音识别系统中的表现并不理想。我们提出的方法利用了已有深度前馈神经网络模型中已经学习到的后验概率分布知识，将其以柔和目标的形式加入到目标函数中。实验证明该方法可以大幅提高 HMM-LSTM 混合模型的识别率。

最后，我们实现了一个跨平台的在线语音识别解码器。该解码器可以支持在多个平台上实时地使用 HMM 混合神经网络模型进行在线大规模连续语音识别功能。解码器支持在高性能服务器上并发地为多个客户端提供语音识别服务，也可以使用近似方法让裁剪过的模型在嵌入式设备上满足实时性要求。

关键词：语音识别；深度学习；神经网络；解码器

Abstract

Automatic speech recognition (ASR) is a key technique for human computer interaction(HCI) and has been widely used in daily life. Applications of deep neural networks in ASR improve the recognition performance of existing ASR systems significantly, which make this technique more usable. However, factors like environment noises impact the performance so much and block further uses heavily. So it's meaningful for us to use neural networks better in ASR.

We focuses on how to use deep learning methods better in large vocabulary continuous speech recognition (LVCSR) to improve the performance and speed up the calculation speed. First, We proposed a method to prune weights in deep neural networks in HMM hybrid models for acoustic modeling in LVCSR. Network pruning methods can decrease the size of the model and make the model more generalizable. This method aims to minimize the change of the objective function with diagonal Hessian matrix assumption. Experiments show that this method does better than pruning weights by their magnitude directly which is used by other researchers.

Next, a method which use existing deep neural network models to train a recurrent neural network (RNN) is proposed. RNN is able to memorize long temporal context, which is suitable for time sequential speech signals, but it doesn't perform well in hybrid models in LVCSR. We exploit the knowledge learned by existing deep neural networks and use it as a soft target in objective function in RNN training. Results show that it can improve the recognition rate for HMM-LSTM hybrid model significantly.

Finally we implement a cross platform decoder for on-line speech recognition. The decoder use HMM neural network hybrid models to do real time decoding with weighted finite state transducers. It can run well in high performance servers with ability to handle multi connections from client or be used as a module in embedded systems efficiently with pruned model and approximation methods.

Key words: automatic speech recognition; deep learning; neural network; decoder

目 录

第 1 章 引言 1
1.1 研究背景与意义 1
1.2 研究现状 2
1.3 研究内容与工作重点 4
1.4 论文结构 4
第 2 章 背景知识介绍 6
2.1 引言 6
2.2 语音识别系统 6
2.2.1 语音识别原理 6
2.2.2 语音识别系统结构 6
2.2.3 基础概念 7
2.2.4 特征提取 8
2.2.5 声学模型 10
2.2.5.1 高斯混合模型 10
2.2.5.2 隐马尔可夫模型 12
2.2.5.3 区分性训练 13
2.2.6 语言模型 15
2.2.7 解码器 16
2.3 深度神经网络 17
2.3.1 神经网络结构 17
2.3.2 神经网络训练 18
2.3.3 使用神经网络进行特征学习 21
2.3.4 HMM-DNN 混合模型 21
2.4 本章小结 22
第 3 章 语音识别中的稀疏神经网络 23
3.1 引言 23
3.2 神经网络中的稀疏化方法 24
3.2.1 目标函数正则化方法 24
3.2.2 Rectifier 激励函数 24
3.2.3 Dropout 方法 25

3.2.4 神经网路权重裁剪 26
3.3 OBD 深度神经网络权重裁剪 27
3.3.1 裁剪过程 27
3.3.2 Hessian 矩阵的反向误差传递 28
3.3.3 实际应用 28
3.4 实验环境说明 28
3.4.1 数据集 28
3.4.2 实验设置 29
3.5 实验结果与分析 29
3.6 本章小结 31
第 4 章 使用隐藏知识训练时间递归神经网络 33
4.1 引言 33
4.2 时间递归神经网络 34
4.2.1 基本概念 34
4.2.2 语音识别中 HMM-RNN 混合模型 36
4.3 深度神经网络中隐藏的知识 37
4.4 隐藏知识迁移训练 38
4.5 实验结果与分析 38
4.5.1 数据库 38
4.5.2 实验设置 39
4.5.3 实验结果 40
4.6 本章小结 41
第 5 章 跨平台在线语音识别系统实现 43
5.1 引言 43
5.2 识别模块 43
5.2.1 特征提取 44
5.2.2 网络计算 44
5.2.3 路径搜索 45
5.3 系统结构 46
5.4 系统演示 47
5.5 本章小结 48

第 6 章 总结与展望 49
6.1 本文工作总结 49
6.2 未来工作展望 50
参考文献 51
致 谢 56
声 明 57
个人简历、在学期间发表的学术论文与研究成果 58

主要符号对照表

ASR	语音识别 (Automatic Speech Recognition)
HMM	隐马尔可夫模型 (Hidden Markov Model)
GMM	高斯混合模型 (Gaussian Mixture Model)
DNN	深度神经网络 (Deep Neural Network)
CNN	卷积神经网络 (Convolutional Neural Network)
RNN	时间递归神经网络 (Recurrent Neural Network)
DCT	离散余弦变换 (Discrete Cosine Transform)
MFCC	梅尔频率倒谱系数 (Mel-Frequency Cepstral Coefficient)
Fbank	梅尔频率滤波器组 (Mel Frequency Filter Bank)

第 1 章 引言

1.1 研究背景与意义

语音识别 (Automatic Speech Recognition, ASR) 是一项研究如何将人类说话的声音识别转换为文本的技术。能够让机器和人类之间自然的交互一直是人机交互 (Human Computer Interaction, HCI) 领域一个重要的研究目标。在当今人类日常生活中，手机，平板，可穿戴设备等各种各样的小型设备得到广泛大量地应用，家中的各种传统电器也愈加智能化。此时，传统的键盘鼠标等输入方式不用多说，就连相对自然一些的触摸等输入方式在越来越小的设备上及需要远距离控制的家电等场景下也显得捉襟见肘。与此相比，语音作为人类非常自然的一种信息交流与信息传递方式在这些情况下却是一种很合适的选择，不仅对空间的需求小，而且对人来说显得更自然，降低了用户的使用门槛，优势可以得到很大程度的体现。这使得语音识别成为当前人机交互非常重要的一环。同时，在多媒体信息检索 (Multimedia Information Retrieval, MMIR) 等领域，语音识别能够帮助人们更好的获取声音中所包含的信息，是它们不可或缺的重要组成部分。

语音识别的重要的实用意义驱使工业界投入大量资源研究相关技术。1997 年，IBM 公司推出了商业语音识别软件 ViaVoice，让更多人体验到了这一便捷的输入方式。2011 年由 Apple 公司和 Nuance 公司合作联合推出的搭载在 iPhone 手机上的 Siri 语音助手服务开启了语音识别技术大范围的普及应用的大门。随后，包括 Google, Microsoft 等各大国际公司，百度，腾讯等中国互联网公司，科大讯飞等专注于语音服务的公司等等纷纷推出了自己的在线语音识别服务和产品。在服务提供商和设备厂商的共同推动下，现在几乎身边新推出的所有移动设备都会包含这一功能。

语音识别不仅具有很强的实用价值，同时还有很重要的研究理论意义。语音信号所具有的时变性、信息量大、非常容易受外界环境影响等特点对模型建立方法和快速计算有着极高的要求。早在 1932 年，贝尔实验室的研究员们就开始了早期语音认知的研究。语音识别的研究从说话人相关、孤立词或关键词识别开始，使用动态时间规整 (Dynamic Time Warping, DTW) 和线性预测编码 (Linear predictive coding, LPC) 等方法。随着研究的进一步推进，隐马尔可夫模型 (Hidden Markov Model, HMM)[1] 的广泛应用给说话人无关的大规模连续语音识别 (Large Vocabulary Continuous Speech Recognition, LVCSR) 的实现提供了可能。在很长的

一段时间内，使用 HMM - 高斯混合模型 (Gaussian Mixture Model, GMM) 建模的声学模型一直是效果最好的建模方式。而 Li Deng 和 Hinton 等人开始尝试深度学习 (Deep Learning, DL) 在语音识别领域的应用[2] 打破了这一僵局，基于深度神经网络 (Deep Neural Network, DNN) 的声学模型大幅度提高了识别率。随后人们快速地转向研究这一技术。

尽管深度学习方法在语音识别上的应用对语音识别技术的发展贡献很大，语音识别系统的识别率相比之前有了很大的提升，但是目前仍然存在着很多未知的问题有待解决。对于在静音环境下正常录制的语音，当前主流语音识别系统已经可以较为准确地进行识别，识别率可以达到 96% 以上，接近人类进行人工语音识别的准确程度[3]；而在实际应用场景下，语音很容易混入环境噪声，回声等干扰因素，不同的麦克风录制获得的语音差别也非常巨大。这些都可能导致语音识别系统的性能急剧下降。这很大程度上阻碍了语音识别在日常生活中的进一步应用。所以，研究如何在语音识别中更好地应用深度学习技术是当前语音识别技术研究领域一个非常重要的课题。

1.2 研究现状

在前深度神经网络年代，处于领先水平的大规模连续语音识别系统通常会使用 HMM-GMM 构建声学模型。在此基础上，研究者们主要从以下几个方面对声学模型进行扩展，提高系统性能：

Haeb-Umbach 等人提出在声学模型中使用线性区分分析 (Linear Discriminant Analysis, LDA) 的方法[4] 增强特征的区分能力，大幅提升了大规模连续语音识别系统的性能。

Leggetter 等人提出使用最大似然线性回归 (Maximum Likelihood Linear Regression, MLLR)[5] 方法进行说话人适应 (Speaker Adaption)，首先构造一个初始的说话人无关的声学模型，然后通过计算线性回归变换的方式更新模型参数或者将变化应用于特征上[6]，提高声学模型对新说话人的适应能力。

在不改变声学模型的基础上，人们仍然可以在特征中使用神经网络的结构优化系统性能。Hermansky 等人通过使用 tandem 特征的方式[7]，将神经网络输出的后验概率作为特征输入 HMM-GMM 模型中训练，大幅度地提高了系统的性能；而 Konig 等人提出使用一种类似瓶颈 (bottleneck) 结构的神经网络，将网络中的投影输出作为特征[8]，同样也取得了很好的效果。

Povey 等人提出利用最大互信息 (Maximum Mutual Information, MMI)[9]，以及最小音素错误 (Minimum Phoneme Error, MPE)[10] 估计的方法，在目标函数中加入

区分正确标注和其他可能的识别结果的准则，进行区分性训练，增强模型[11] 或特征[12] 的区分能力。在大数据集上实验的证明区分性训练的方法可以有效地提高系统的识别率。

Povey 等人还提出了子空间高斯混合模型 (Subspace Gaussian Mixture Model, SGMM)[13] 用于声学模型建模。该方法使各个状态共享一个相同的高斯混合模型结构，但是均值和混合权重可以在一个统一的参数子空间之内变化。这样的声学模型表达更加紧凑，相比传统的建模方式可以获得更好的效果，特别是在小数据集上。

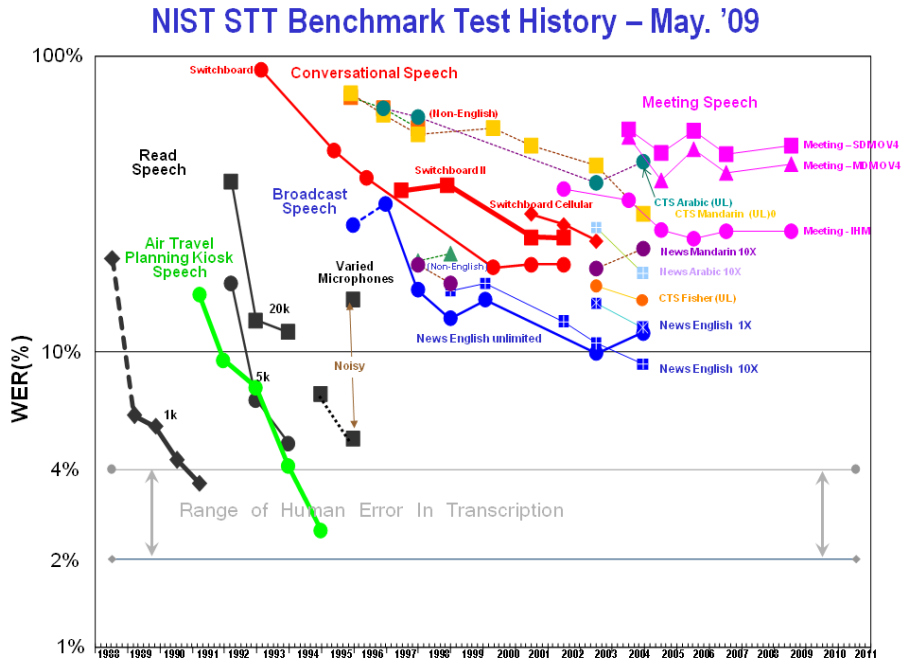

图 1.1　语音识别系统发展历程

来源：NIST 语音识别评估历史，http://itl.nist.gov/iad/mig/publications/ASRhistory/index.html

在大家都纷纷转向使用 HMM-DNN 混合模型作为声学模型之后，研究者关于神经网络的研究包括以下几个方面：

人们主要使用多个受限玻尔兹曼机 (Restricted Boltzmann Machine, RBM) 堆叠构成深度置信网络 (Deep Belief Network, DBN) 来进行深度神经网络的逐层预训练[14][2][15][16]，改善层数过多时网络易陷入局部最优的问题。

Dahl 等人提出在深度神经网络中应用线性输出单元 (rectified linear unit) 和 dropout 技术[17]，改善神经网络过拟合的问题，提高系统性能；Yajie 等人在深度神经网络中应用 maxout 技术[18]，进一步提高了在小数据集上的系统性能。

Abdel 等人成功地将卷积神经网络 (Convolutional Neural Network, CNN) 应用到 HMM-NN 混合模型中，在频域通过 pooling 的方法归一化不同说话人的差

别[19]，提高了系统的性能。

Vesely 等人在 HMM-DNN 混合模型中成功应用了序列区分性训练的方法[20]，使用状态级最小贝叶斯风险 (state level Minimum Bayes Risk, sMBR) 准则在 RM 数据集上取得了非常好的效果。

1.3 研究内容与工作重点

本文我们主要研究如何在基于 HMM 混合模型的大规模连续语音识别系统中更好的应用深度学习方法，包括以下几个内容:

1. 提出了一种深度神经网络权重裁剪方法

关于网络权重裁剪这种稀疏化方法，我们提出了一种在语音识别 HMM 混合系统中的深度神经网络权重裁剪方法。该方法以对角化 Hessian 矩阵假设为基础，以最小化目标函数变化为目标对深度神经网络进行权重裁剪。我们讨论了这种方法在语音识别中使用的意义，并将其与另外一种语音识别中的深度神经网络权重裁剪方法进行了对比。结果表明，我们的方法对深度神经网络的权重裁剪效果更好，能够在相同的稀疏度下获得更好的性能。

2. 提出了一种利用深度前馈神经网络迁移训练时间递归神经网络的方法

时间递归神经网络由于具备长时间上下文的记忆能力，非常适合对语音这种序列化的向量建模。但是，在很长的一段时间内，直接将时间递归神经网络应用在语音识别中的 HMM 混合模型中无法获得很好的效果。我们尝试探索这一问题，进一步讨论了网络训练目标的不匹配问题，提出了一种利用在 HMM 混合模型中已经训练好的深度前馈神经网络的隐藏知识迁移训练时间递归神经网络的方法。实验结果表明该方法可以大幅度改善小规模数据上 HMM-LSTM 混合模型的性能，获得了比基线深度前馈神经网络更好的效果。

3. 实现了一个跨平台的在线语音识别系统

我们实现了一个跨平台的在线语音识别系统，可以支持在多个平台下使用 HMM 混合神经网络模型进行在线解码功能。针对嵌入式设备设备端上的非联机识别任务，我们应用并实现了几种裁剪和近似方法，使其可以满足实时性的要求。我们实现了一个高效的语音识别服务服务器端，支持客户端并发地使用在线语言识别服务。

1.4 论文结构

第2章介绍语音识别系统的基础知识和各个基本模块的原理，回顾了神经网络的基本概念，为后面的章节打下理论基础。

第3章主要介绍和深度神经网络中稀疏化方法相关的内容，着重阐述网络权重裁剪这种方法在语音识别 HMM 混合网络模型中应用相关的内容。

第4章和时间递归神经网络相关，讲述了我们如何利用已有的深度前馈神经网络模型更好地在大规模连续语音识别中应用时间递归神经网络。

第5章详细地描述了我们的跨平台在线语音识别系统的实现方案和使用到的实用技术。

第6章对论文关于在语音识别中应用深度学习的方法进行了总结，并对论文中提到的各项方法的后续工作进行了讨论和展望。

第 2 章　背景知识介绍

2.1　引言

本章中我们给出语音识别相关的基础知识，介绍语音识别系统的各个组成部分及相关原理，并对神经网络的基本概念和原理作出说明。

2.2　语音识别系统

2.2.1　语音识别原理

在这里我们重新给出语音识别问题的形式化定义。对于给定的一段语音信号 s，语音识别的目标是求出与之对应的最有可能的输出序列 $\hat{W}$，即

$$\hat{W} = \arg\max_{W} P(W|s) \tag{2-1}$$

通过贝叶斯公式，我们可以将2-1变形为

$$\hat{W} = \arg\max_{W} \frac{P(s|W)P(W)}{P(s)} \tag{2-2}$$

由于在时域上的信号 s 不好直接进行各种处理，所以我们通常会通过一次特征提取 (Feature extraction) 的过程将其分段计算转换为频域上滤波后的特征向量序列 X。考虑到在一次识别过程中 X 是固定的，我们可以得到语音识别问题的一般定义形式

$$\hat{W} = \arg\max_{W} P(X|W)P(W) \tag{2-3}$$

此时，$P(X|W)$ 表征特征向量序列 X 符合输出序列 W 的概率，我们称用于估算该概率的模型为声学模型 (Acoustic Model, AM) 。$P(W)$ 表征输出序列 W 符合当前语言特性的概率，我们称该模型为语言模型 (Language Model, LM) 。

2.2.2　语音识别系统结构

一个典型的语音识别系统如下图2.1所示。

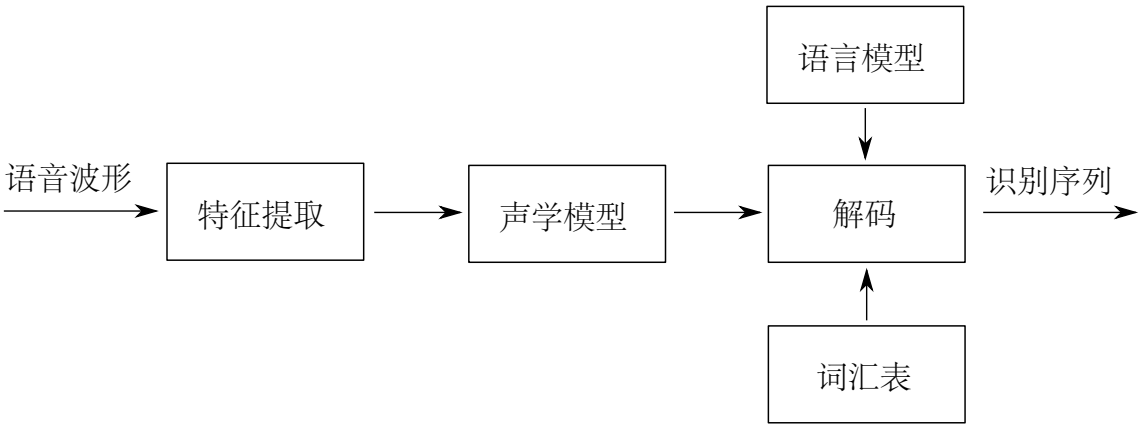

图 2.1　语音识别系统

我们可以看到，语音识别系统包含如下四个模块：特征提取、声学模型、语言模型和解码。其中，特征提取模块以语音的声音信号作为输入，将声音信号从时域转换到频域，为声学模型提取合适的特征向量；声学模型以特征提取模块输出的特征向量作为输入，根据声学特性为输入计算一个声学模型的分数；语言模型根据语言学上的信息，计算一句话对应的词序列的可能性；解码器模块组合了声学模型、语言模型和词表的信息，输出对于输入的特征向量最有可能词序列。为了后面叙述方便，我们先给出和语音识别相关的几个基础概念，然后再逐模块介绍整个语音识别系统的工作流程。

2.2.3　基础概念

在当前人们的认知中，语音是一段连续的声音流，从长时间来看一直在不同的状态之间切换，而从一段比较短的时间来看可以被认为处于一个相对稳定的状态之中。我们认为语音符合短时平稳假设，这是后文工作信号分析处理部分的基础。从语音学的角度来说，字或词可以被认为由多个控制节奏、韵律、轻重音的音节 (syllable) 所组成，而音节又拆分成多个可以区分意义的单元。我们称这样的最小单元为音素 (phoneme)。英语下一个具体的例子为：

单词 English

音节 Eng lish

音素 ɪ ŋ g l ɪ ʃ

而从语音识别的角度来看，一个音素又可以被拆分为多个子状态。根据上下文的不同，同一个音素在发音时也有区别，所以我们还会考虑这种上下文相关的音素，称为上下文相关音素 (context dependent phone)。一种常见的上下文相关音素选取前后各一个音素作为上下文，称为三音素 (triphone)。主流的语音识别模型选择建模的单元为上下文相关音素子状态 (context dependent state)。

完全独立的上下文相关音素子状态数量过于庞大（一种常见的英语音素划分包括 48 个音素，对于三状态三音素即 $3 \times 48^3 = 331776$ 个子状态），同时数据也常常无法覆盖到全部情况。对于这样的问题，我们可以通过共享相近的上下文相关音素子状态来解决，称为共享上下文相关音素子状态 (shared context dependent state)。我们可以通过构建一棵决策树来决定如何共享。

语音识别性能的评估主要通过考察系统在固定数据集上的准确率的方式来完成。测试集中会有一定数量的语音片段 (utterance) 和其所对应的人工给出的文本标注 (transcription)。系统会输出对于语音片段的一个最优预测（hypothesis），我们通过对比预测和标注的方式衡量系统性能。语音词错误率 (Word Error Rate, WER) 是最常见的衡量语音识别系统性能的指标，定义为

$$WER = \frac{S + D + I}{N} \tag{2-4}$$

其中，

- S 为发生替换 (substitution) 错误的词的数量
- D 为发生删除 (deletion) 错误的词的数量
- I 为发生插入 (insertion) 错误的词的数量
- N 为标注中全部词的数量

而衡量语音识别系统效率的常用指标是实时率 (Real Time Factor, RTF)，定义为处理一段语音的时间与语音总时间长度的比值。当一个语音识别系统的实时率小于 1 的时候，系统可以不随着时间的推移在缓冲区中持续堆积语音无法处理完成，我们称这样的情况为达到实时性要求。

2.2.4　特征提取

特征提取部分的目的是提取中语音信号中和人说话相关的部分信息，供声学模型对语音更方便合理的建模。首先，我们使用预加重 (Pre-emphasis) 技术平衡语音的高低频分量。考虑到语音信号的短时平稳特性，我们接下来对语音进行分帧 (Frame) 处理，对每一帧作短时分析，再将语音时域信号通过离散傅里叶变换 (Discrete Fourier Transform, DFT) 转换到频域。我们一般会仿照人耳的的响应对声音作滤波操作。对人耳的听觉特征的研究工作表明，人耳对声音的敏感程度随着频率的升高而逐渐加速下降。在这里最常用的滤波器是由三角带通滤波器组成的

梅尔滤波器，可用2-5表示。

$$H_m(k) = \begin{cases} 0 & k < f(m-1) \\ \dfrac{k - f(m-1)}{f(m) - f(m-1)} & f(m-1) \le k < f(m) \\ \dfrac{f(m+1) - k}{fm + 1 - f(m)} & f(m) \le k < f(m+1) \\ 0 & k >= f(m+1) \end{cases} \tag{2-5}$$

其中, $f(m-1)$, $f(m)$, $f(m+1)$ 分别为第 m 个三角带通滤波器的下限，中心，上限频率，f 的计算公式为

$$f(m) = Mel^{-1}[Mel(f_l) + m\frac{Mel(f_h) - Mel(f_l)}{M+1}] \tag{2-6}$$

梅尔频率 (Mel frequency) 与频率的换算公式为

$$Mel(f) = 1125\ln(\frac{f}{700} + 1) \tag{2-7}$$

$$Mel^{-1}(f) = 700(\exp(\frac{f}{1125}) - 1) \tag{2-8}$$

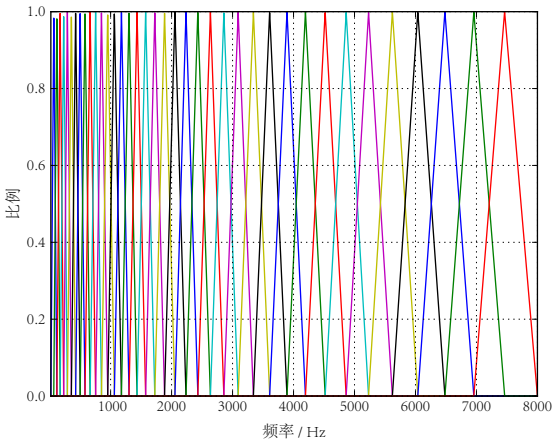

图 2.2　梅尔频率滤波器组

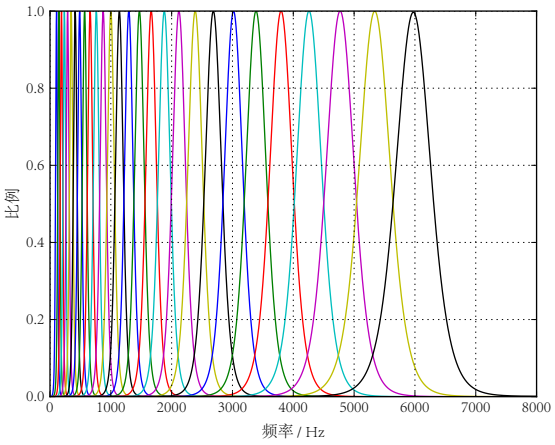

图 2.3　伽玛音调滤波器组

针对不同的采样率 f_s，我们通常会取不同的截止频率 f_h, 截断频率 f_l 和 M。一个典型的取值方案是 $f_h = 16000, f_l = 20, M = 40$。经过梅尔滤波器滤波后的的 M 个能量值可以取 log 后直接作为一组 M 维的特征向量，称为梅尔频率滤波器组 (Mel Frequency Filter Bank, Fbank) 特征。而对于传统声学模型，我们通常还需要再多进行一步离散余弦变换 (Discrete Cosine Transform, DCT) 操作，以达到去相关性的效果。经过梅尔滤波器和 DCT 变换后的特征即为被广泛应用的梅尔频率倒谱系数 (Mel-Frequency Cepstral Coefficient, MFCC) 特征。

除了梅尔频率滤波器组，我们还可以使用更精确表征人耳听觉特性的感知线性预测 (Perceptual Linear Prediction, PLP) 特征[21] 或使用伽玛音调滤波器 (Gamma Tone Filter, GTF) 。研究 [22] 表明，使用 GTF 的伽玛音调滤波器倒谱系数 (Gamma tone Filter Cepstral Coefficient, GFCC) 特征具有对噪音更鲁棒的效果。

2.2.5　声学模型

声学模型是语音识别系统中非常重要的一个组件，对不同基本单元的区分能力直接关系到识别结果的好坏。现在主流的 LVCSR 系统基本都使用隐马尔可夫模型来刻画不同的语音基本单元之间的过渡关系，而用于描述或区分短时语音基本单元静态特征的模型则由从使用高斯混合模型逐渐全面转向使用 DNN 或更复杂的神经网络如卷积神经网络 (Convolutional Neural Network, CNN) 和时间递归神经网络 (Recurrent Neural Network, RNN) 等。接下来我们首先对高斯混合模型和隐马尔可夫模型进行介绍，而在后面的章节中再详细说明和神经网络相关的内容。

2.2.5.1　高斯混合模型

对于一个随机向量 $\boldsymbol{x}$，如果它的联合概率密度函数符合公式2-9，则称它服从高斯分布，并记为 $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$。

$$p(\boldsymbol{x}) = \frac{1}{(2\pi)^{D/2}|\boldsymbol{\Sigma}|^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})] \tag{2-9}$$

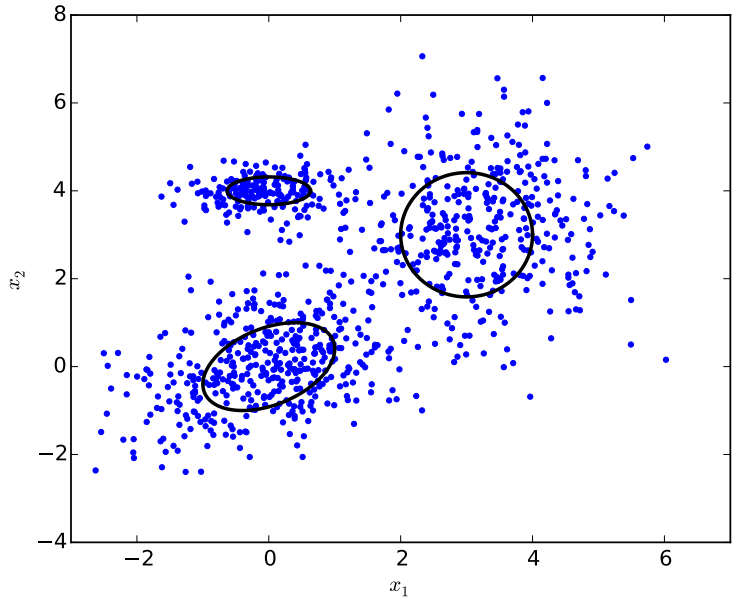

图 2.4　使用高斯混合模型对二维数据建模

其中，$\boldsymbol{\mu}$ 为分布的期望，$\boldsymbol{\Sigma}$ 为分布的协方差矩阵。

高斯分布有很强的近似真实世界数据的能力，同时又易于计算，因此被广泛地应用在各个学科之中。但是，仍然有很多类型的数据不好被一个高斯分布所描述。这时候我们可以使用多个高斯分布的混合分布来描述这些数据，由多个分量分别负责不同潜在的数据来源。此时，随机变量符合密度函数

$$p(\boldsymbol{x}) = \sum_{m=1}^{M} \frac{c_m}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_m|^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_m)^T\boldsymbol{\Sigma}_m^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_m)] \tag{2-10}$$

其中，M 为分量的个数，通常由问题规模来确定。

我们称认为数据服从混合高斯分布所使用的模型为高斯混合模型。高斯混合模型被广泛的应用在很多语音识别系统的声学模型中。考虑到在语音识别中向量的维数相对较大，所以我们通常会假设混合高斯分布中的协方差矩阵 $\boldsymbol{\Sigma}_m$ 为对角矩阵。这样既大大减少了参数的数量，同时可以提高计算的效率。

我们可以利用最大似然估计 (Maximum Likelihood Estimation, MLE) 的方法来获得高斯混合模型的参数。在语音识别中我们通常使用的 EM(expectation maximization) 算法来计算模型参数。该算法迭代地进行计算期望的 E 步骤和作最大化的 M 步骤。

使用高斯混合模型对短时特征向量建模有以下几个好处：首先，高斯混合模型的具有很强的建模能力，只要分量总数足够多，高斯混合模型就可以以任意精度来逼近一个概率分布函数；另外，使用 EM 算法可以很容易地使模型在训练数据上收敛。对于计算速度和过拟合等问题，人们还研究出了参数绑定的 GMM 和子空间高斯混合模型 (subspace GMM) 来解决。除了使用 EM 算法作最大似然估计以外，我们还可以使用和词或音素错误率直接相关的区分性的误差函数来训练高斯混合模型，能够极大地提高系统性能。因此，直到在声学模型中使用深度神经网络的技术出现之前，高斯混合模型一直是短时特征向量建模的不二选择。

但是，高斯混合模型同样具有一个严重的缺点：高斯混合模型对于靠近向量空间上一个非线性流形 (manifold) 上的数据建模能力非常差。例如，假设一些数据分布在一个球面两侧，且距离球面非常近。如果使用一个合适的分类模型，我们可能只需要很少的参数就可以将球面两侧的数据区分开。但是，如果使用高斯混合模型描绘他们的实际分布情况，我们需要非常多的高斯分布分量才能足够精确地刻画。这驱使我们寻找一个能够更有效利用语音信息进行分类的模型。

2.2.5.2　隐马尔可夫模型

我们现在考虑一个离散的随机序列，若转移概率符合马尔可夫性质，即将来状态和过去状态独立，则称其为一条马尔可夫链 (Markov Chain)。若转移概率和时间无关，则称其为齐次 (homogeneous) 马尔可夫链。马尔可夫链的输出和预先定义好的状态一一对应，对于任意给定的状态，输出是可观测的，没有随机性。如果我们对输出进行扩展，使马尔可夫链的每个状态输出为一个概率分布函数。这样的话马尔可夫链的状态不能被直接观测到，只能通过受状态变化影响的符合概率分布的其他变量来推测。我们称以这种以隐马尔可夫序列假设来建模数据的模型为隐马尔可夫模型。隐马尔可夫模型包含以下几个基本元素：

1. 状态

状态空间由一个有限的离散集合组成，表示为 $q_t \in \{s^{(j)}, j = 1, 2, \cdots, N\}$。其中 N 为总状态数。

2. 转移概率

转移矩阵可表示为 $A = [a_{i,j}],\ \ i, j = 1, 2, \cdots, N$。矩阵中的元素满足

$$a_{i,j} = P(q_t = j \mid q_{t-1} = i), \quad i, j = 1, 2, \cdots, N \tag{2-11}$$

3. 初始状态概率

即在最初状态时，模型处于每个状态的概率，$\pi_i = P(q_1 = i)$。可表示为

$$\boldsymbol{\pi} = [\pi_i],\ i = 1, 2, \cdots, N,\ \sum_{i}^{N} \pi_i = 1 \tag{2-12}$$

4. 观测概率分布

即处于不同状态时，观测输出的概率分布函数 $P(\boldsymbol{o}_t \mid s^{(i)}),\ i = 1, 2, \cdots, N$。若观测 $\boldsymbol{o}_t$ 是离散的，那么概率分布函数由每个观测符号 $\{\boldsymbol{v}_1, \boldsymbol{v}_2, \cdots, \boldsymbol{v}_K\}$ 的概率给出

$$b_i(k) = P[\boldsymbol{o}_t = \boldsymbol{v}_k \mid q_t = i],\ i = 1, 2, \cdots, N \tag{2-13}$$

而在语音识别中最常用的观测概率分布是上文中公式2-10中提到的连续的概率分布函数高斯混合分布

$$b_i(\boldsymbol{o}_t) = \sum_{m=1}^{M} \frac{c_{i,m}}{(2\pi)^{D/2}|\boldsymbol{\Sigma}_{i,m}|^{1/2}} \exp[-\frac{1}{2}(\boldsymbol{o}_t - \boldsymbol{\mu}_{i,m})^T \boldsymbol{\Sigma}_{i,m}^{-1}(\boldsymbol{o}_t - \boldsymbol{\mu}_{i,m})] \tag{2-14}$$

一个一阶隐马尔可夫模型的结构如图2.5所示。对应到语音识别系统中，我们使用隐马尔可夫模型来刻画一个音素内部子状态变化，来解决特征序列到多个语音基本单元之间对应关系的问题。

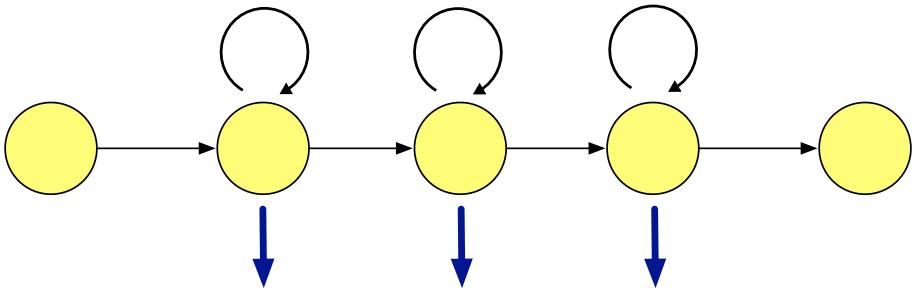

图 2.5　常用的一阶隐马尔可夫模型

在语音识别任务中使用隐马尔可夫模型需要计算模型在一段语音片段上的可能性。对于一个有限长度状态序列 $\boldsymbol{q}_1^T = (q_1, q_2, \cdots, q_T)$，以及观测输出序列 $\boldsymbol{o}_1^T = \boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_T$，我们可以记 $P((o)_1^T, \boldsymbol{q}_o^T)$ 为观测序列和状态序列联合可能性，$P((o)_1^T \mid \boldsymbol{q}_o^T)$ 为在状态序列 $\boldsymbol{q}_1^T$ 条件下的观测序列输出的可能性。

在 HMM-GMM 混合系统中，条件可能性 $P((o)_1^T \mid \boldsymbol{q}_o^T)$ 可以表示为

$$P((o)_1^T \mid \boldsymbol{q}_o^T) = \prod_{t=1}^{T} b_t(\boldsymbol{o}_t) \tag{2-15}$$

指定状态序列 $\boldsymbol{q}_1^T$ 的概率为各个转移概率的乘积，即

$$P(\boldsymbol{q}_1^T) = \pi_{q_1} \prod_{t=1}^{T-1} a_{q_t q_{t+1}} \tag{2-16}$$

在进行识别的时候，我们会采用 Viterbi 算法寻找一个最可能的状态序列，这时只需要计算联合概率即可：

$$P(\boldsymbol{o}_1^T, \boldsymbol{q}_1^T) = P((o)_1^T \mid \boldsymbol{q}_o^T) P(\boldsymbol{q}_1^T) \tag{2-17}$$

而在训练的时候，我们需要使用 Baum-Welch 算法[23] 学习隐马尔可夫模型参数，进行最大似然估计 (Maximum Likelihood Estimation, MLE)。Baum-Welch 算法是 EM (Expectation-Maximization) 算法[24] 的一种特例，利用前后项概率信息迭代地依次进行计算条件期望的 E 步骤和最大化条件期望的 M 步骤。

2.2.5.3　区分性训练

在语音识别中使用最大似然估计方法估算模型参数，对于某一句特定的语音片段，模型将被调整为最大化该片段的标注对应的词序列的可能性，但是并没有考虑其他可能的词序列可能性。因此，最大似然估计方法的训练目标和系统的性能指标并不一致，可能会存在训练目标优化地更好，但语音识别系统的识别性能反而下降的情况。针对这这个问题，研究者提出了区分性训练 (discriminative

training)[25][26][27] 的概念，考虑潜在的与正确标注竞争的词序列，尝试减少错误的识别结果的概率。最初区分性训练在小规模词表识别任务中取得了一些效果，但是无法直接应用于大规模连续语音识别中。原因包括可竞争的词序列数量过多影响训练，以及受训练数据和模型规模所限泛化能力不够强等。随着标注数据逐渐积累越来越多，以及计算能力的大幅提高，在大规模连续语音识别中的区分性训练方法变得更加现实，同时相比于仅使用最大似然估计方法的语音识别系统可以达到更好的识别的性能。

考虑 R 个观测序列 $\{\boldsymbol{o}_1, \boldsymbol{o}_2, \cdots, \boldsymbol{o}_R\}$ 及对应的标注 $\{w_1, w_2, \cdots, w_R\}$。我们将每个标注对应的 HMM 模型的复合记为 $\{M_1, M_2, \cdots, M_R\}$，则有最大似然估计方法可以表示为

$$f_{MLE}(\lambda) = \sum_{r=1}^{R} \log P_\lambda(\boldsymbol{o}_r|M_r) \tag{2-18}$$

最大互信息 (Maximum Mutual Information, MMI) 准则的目标是最大化观测序列和词序列之间的互信息，可以表示为

$$f_{MMI}(\lambda) = \sum_{r=1}^{R} \log \frac{P_\lambda(\boldsymbol{o}_r|M_r)P(w_r)}{\sum_{\hat{w}} P_\lambda(\boldsymbol{o}_r|M_{\hat{w}})P(\hat{w})} \tag{2-19}$$

其中，λ 为声学模型的参数，$P(w)$ 为语言模型给出的词序列的概率，分母可以记为

$$P_\lambda(\boldsymbol{o}_r|M_{den}) = \sum_{\hat{w}} P_\lambda(\boldsymbol{o}_r|M_{\hat{w}})P(\hat{w}) \tag{2-20}$$

从上式中我们可以看出，优化最大互信息准则也就是在最大化统最大似然准则的情况下最小化考虑所有可能情况的分母。但是，在大规模连续语音识别场景下，分母包含所有可能的情况，在大规模的声学模型和语言模型中几乎没有可能计算。所以我们采取近似的方法减少分母部分的计算量。

常用的近似方法利用解码网格 (lattice) 的信息来作近似[28]。网格是一种语音识别系统解码器输出的图结构。和最直接的得到唯一最优结果的输出不同，我们可以在解码的过程中记录下一定宽度的中间结果信息，包括时间点，声学模型的似然打分，语言模型的似然打分等，在语音片断识别完成后将这些信息保存下来。一个网格的示例如图2.6所示。网格中保存了当前模型对语音片断在一定误差范围内的多种识别结果，很适合用于作上式中分母部分的近似。整体训练过程如下:

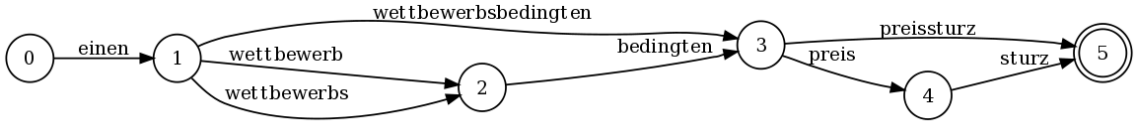

图 2.6　关于词的解码网格示例

来源：Moses 的 lattice 示例，http://www.statmt.org/moses/?n=Moses.WordLattices

1. 使用最大似然估计的方法训练好一个声学模型。
2. 使用一个合适的语言模型配合训练好的声学模型对训练集作解码，生成解码网格。
3. 对解码网格里的每一条边，通过声学模型生成 HMM 序列和 Viterbi 解码片断端点信息。
4. 计算每一个状态高斯部分的概率。
5. 优化区分性训练目标函数。我们通常使用扩展的 Baum-Welch 算法对区分性训练的目标函数作优化。

其他的用于区分性训练的准则还包括：

增强的最大互信息 (boosted MMI, BMMI)[11]，表示为

$$f_{BMMI}(\lambda) = \sum_{r=1}^{R} \log \frac{P_\lambda(\boldsymbol{o}_r|M_r)P(w_r)}{\sum_{\hat{w}} P_\lambda(\boldsymbol{o}_r|M_{\hat{w}})P(\hat{w})e^{-bA(\hat{w},w_r)}} \tag{2-21}$$

其中，b 为增强系数，可以取为 0.5；$A(w,\hat{w})$ 为正确的标注和另一个识别结果之间差异的度量值，从词、音素或状态来计算均可。

BMMI 在原始 MMI 的基础上增强了对那些和语音标注相差更大的识别结果的可能性的惩罚，进一步加强了区分的效果。

最小音素错误 (Minimum Phone Error, MPE)[10] 也是一种常见的区分性训练准则，表示为

$$f_{MPE}(\lambda) = \sum_{r=1}^{R} \log \frac{\sum_{\hat{w}} P_\lambda(\boldsymbol{o}_r|M_{\hat{w}})P(\hat{w})A(\hat{w},w_r)}{\sum_{\hat{w}} P_\lambda(\boldsymbol{o}_r|M_{\hat{w}})P(\hat{w})} \tag{2-22}$$

MPE 将分子修改为分母的关于偏差的加权平均，实验表明可以使识别性能得到提高。

2.2.6　语言模型

在很多情况下，不同的词序列的发音可以非常相近，甚至完全一样。比如在中文中，句子 “这个 gong shi 很难推导。” 中的 “gong shi” 人们听到后会理解到是 “公式”，而单单通过发音我们无法确定它到底是 “公式”，“攻势” 或是 “公示”。这

时候我们需要语言模型对输出的词序列建模，刻画一个词序列在该语言环境下的可能性。对于有严格的语法限制的使用场景，如语音命令控制、数字识别等等，我们直接构建该情况对应的固定语法来决定词序列输出的合法性。而对于正常的人类语言，我们需要使用一个统计语言模型对词序列进行建模。

最简单又却又最常用的语言模型是 N 元语言模型 (N-gram Language Model, N-gram LM) 。N 元语言模型假设当前在给定上文环境下，当前词的概率只与前 N-1 个词相关。于是词序列 $w_1, \ldots, w_m$ 的概率 $P(w_1, \ldots, w_m)$ 可以近似为

$$\begin{aligned} P(w_1, \ldots, w_m) &= \prod_{i=1}^{m} P(w_i \mid w_1, \ldots, w_{i-1}) \\ &\approx \prod_{i=1}^{m} P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) \end{aligned} \tag{2-23}$$

例如对于词序列这个 公式 很难 推导，3 元 (tri-gram) 语言模型计算词序列概率为

$$\begin{aligned} P_{tri}(W) &= P(\text{推导}|\text{这个 公式 很难})P(\text{很难}|\text{这个 公式})P(\text{公式}|\text{这个})P(\text{这个}) \\ &\approx P(\text{推导}|\text{公式 很难})P(\text{很难}|\text{这个 公式})P(\text{公式}|\text{这个})P(\text{这个}) \end{aligned} \tag{2-24}$$

为了得到公式中的每一个词在给定上文下的概率，我们需要一定数量的该语言文本来估算。可以直接使用包含上文的词对在全部上文词对中的比例来计算该概率，即

$$P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1}) = \frac{\text{count}(w_{i-(n-1)}, \ldots, w_{i-1}, w_i)}{\text{count}(w_{i-(n-1)}, \ldots, w_{i-1})} \tag{2-25}$$

对于在文本中未出现的词对，我们需要使用平滑方法来进行近似，如 Good-Turing 估计或 Kneser-Ney 平滑等。

由于词表规模很大，所以很多复杂模型无法在计算量、模型大小和实际性能之间取得合理的平衡。但各种神经网络语言模型 (Neural Network Language Model, NNLM) 特别是循环神经网络语言模型 (Recurrent Neural Network Language Model, RNNLM) 逐渐兴起，通过和 N 元语言模型结合可以在语音识别中提高识别率。

2.2.7　解码器

解码器是识别阶段的核心组件，通过训练好的模型对语音进行解码，获得最可能的词序列，或者根据识别中间结果生成识别网格 (lattice) 以供后续组件处理。解码器部分的核心算法是动态规划算法 Viterbi。由于解码空间非常巨大，通常我们在实际应用中会使用限定搜索宽度的令牌传递方法 (token passing)。

传统解码器会完全动态生成解码图 (decode graph)，如著名语音识别工具 HTK(HMM Tool Kit) 中的 HVite 和 HDecode 等。这样的实现内存占用较小，但考

虑到各个组件的复杂性，整个系统的流程繁琐，不方便高效地将语言模型和声学模型结合起来，同时更加难以扩展。现在主流的解码器实现会一定程度上使用预生成的有限状态变换器 (Finite State Transducer, FST) 作为预加载的静态解码图。这里我们可以将语言模型 (G)，词汇表（L），上下文相关信息 (C)，隐马尔可夫模型 (H) 四个部分分别构建为标准的有限状态变换器，再通过标准的有限状态变换器操作将他们组合起来，构建一个从上下文相关音素子状态到词的变换器。这样的实现方法额外使用了一些内存空间，但让解码器的指令序列变得更加整齐，使得一个高效的解码器的构建更加容易。同时，我们可以对预先构建的有限状态变换器进行预优化，合并和剪掉不必要的部分，使得搜索空间变得更加合理。具体的实现细节还会在后面的章节中详细阐述。

2.3　深度神经网络

早在上个世纪八十年代，就有研究者在语言识别中使用神经网络作为分类器[29]。但受限于当时机器的计算能力，语音数据的稀少，以及对语音基本单元建模的选择等等因素，神经网络分类器并没有在后来成为语音识别系统中成为主流，效果不如使用高斯混合模型。但随着新世纪人们对神经网络的重新认识，深度学习的风潮再次席卷了语音界，人们纷纷转向研究深度神经网络在语音识别中的应用。深度神经网络模型是区分性 (discriminative) 的模型，对于区分不同的基本单位这个任务来说，比需要描述完整分布的产生性 (generative) 模型高斯混合模型模型需要的参数相对更少，更容易获得好的效果。

2.3.1　神经网络结构

一个深度神经网络是一个有很多个隐藏层的前馈人工神经网络。其结构如图2.7所示。输入层的维数与特征向量维数相同，隐藏层的维数和层数人工指定，输出层的维数和语音基本单元的个数相同。网络的目标是输出特征向量在各个语音基本单元的后验概率 (Posterior probability)。其公式表示如下:

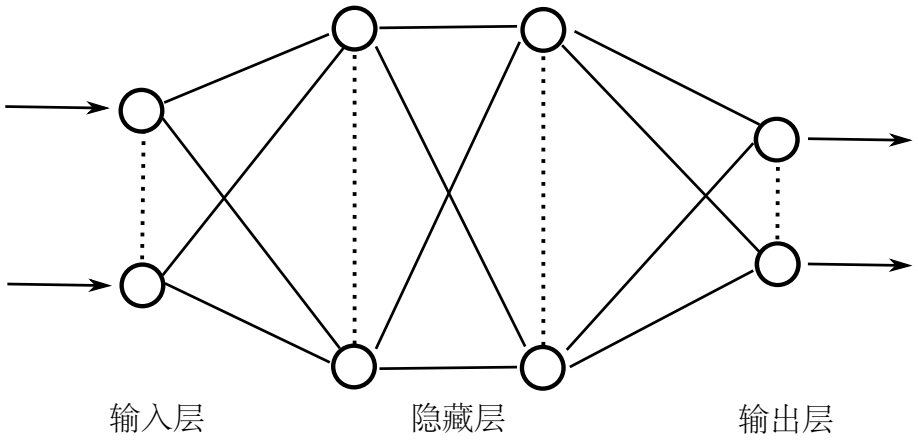

图 2.7　深度神经网络

$$\begin{aligned} \boldsymbol{a_0} &= \sigma(W_0\boldsymbol{x} + \boldsymbol{b}_0) \\ \boldsymbol{a}_m &= \sigma(W_m\boldsymbol{a}_{m-1} + \boldsymbol{b}_m), \quad \text{for } i = m \ldots M-1 \\ \boldsymbol{y} &= \theta(W_M\boldsymbol{a}_{M-1} + \boldsymbol{b}_M) \end{aligned} \tag{2-26}$$

其中，$\boldsymbol{x}$ 表示输入的特征向量，$\boldsymbol{y}$ 表示输出的后验概率，k 为隐含层的个数，$\boldsymbol{a_m}$ 为第 m 个隐层的输出。W_m 为网络的第 m 个仿射变换矩阵，表示为

$$W_m = \begin{bmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,j} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,j} \\ \vdots & \vdots & \ddots & \vdots \\ w_{i,1} & w_{i,2} & \cdots & w_{i,j} \end{bmatrix} \tag{2-27}$$

i 为后一层的维数，j 为前一层的维数，$w_{i,j}$ 为后一层的第 i 个神经元连接前一层第 j 个神经元的权重。b_m 为网络第 m 个仿射变换的偏置向量 $[b_1\ b_2\ \cdots\ b_i]^T$。$\sigma$ 为输入层和隐藏层的激励函数，通常取 sigmoid 函数。

$$\sigma(\boldsymbol{x}) = \frac{1}{1+e^{-\boldsymbol{x}}} \tag{2-28}$$

对于分类器，θ 为输出层的的归一化函数，通常取 softmax 函数。

$$\theta(\boldsymbol{x})_i = \frac{e^{x_i}}{\sum_{k=1}^{K} e^{x_k}}, \quad \text{for } i = 1 \ldots K \tag{2-29}$$

2.3.2　神经网络训练

我们需要通过训练数据对神经网络的参数进行训练。最直接的训练目标就是使训练数据中的每个特征向量的输出误差最小。常见的误差函数包括均方误差

(Mean Square Error, MSE) 和交叉熵 (Cross Entropy, CE)。对于用于分类的神经网络，我们通常使用交叉熵函数，定义为

$$E_{CE}(\boldsymbol{w}) = -\sum_{n=1}^{N}\{\boldsymbol{t}_n \ln \boldsymbol{y}_n + (1-\boldsymbol{t}_n)\ln(1-\boldsymbol{y}_n)\} \tag{2-30}$$

其中，$\boldsymbol{w}$ 为网络参数的总称，N 为训练样本的数量，$\boldsymbol{y}_n$ 为第 n 个样本的网络输出，$\boldsymbol{t}_n$ 为第 n 个样本的真实标注。

我们需要使用最优化方法使误差函数最小化。最常见的最优化方法就是梯度下降法 (Gradient Descent)。由于神经网络的误差函数是连续的，所以我们可以使用梯度下降法求得网络误差函数的局部最优解。梯度下降法可以表示为

$$\boldsymbol{w}^{\tau+1} = \boldsymbol{w}^{\tau} - \eta\nabla E(\boldsymbol{w}^{\tau}) \tag{2-31}$$

其中，τ 为迭代的轮数，η 为学习率 (learning rate)。

一个常见的扩展为在误差函数中使用冲量 (momentum)[30]，式子可以表示为

$$\begin{aligned}\boldsymbol{v}^{\tau+1} &= \mu\boldsymbol{v}^{\tau} - \eta\nabla E(\boldsymbol{w}^{\tau})\\ \boldsymbol{w}^{\tau+1} &= \boldsymbol{w}^{\tau} + \boldsymbol{v}^{\tau+1}\end{aligned} \tag{2-32}$$

使用冲量可以保持之前的下降的速度方向，可以起到加速梯度下降的效果。

而具体的梯度 ∇E 需要分别对不同的隐藏层，输出层部分计算。对于一般的神经网络，我们有通用的反向误差传递 (Back propagation, BP) 算法可以通过将误差从输出层逐层向前传递的方法求得误差函数对于每一个参数矩阵的梯度。其算法流程如下:

对于训练数据中的每个样本 $\boldsymbol{x}_i$，首先我们按照网络的定义通过前向传递 (forward pass) 计算当前网络对于该输入的输出 $\boldsymbol{y}_i$，并根据选择的误差函数计算与数据标注 $\boldsymbol{t}_i$ 的误差 E。接下来，我们逐层为每个隐藏层和输入层对应的仿射矩阵的计算梯度 $\nabla E(\boldsymbol{W}_i)$。最后我们根据计算好的梯度使用梯度下降法更新网络的各个权重矩阵，并重复上述过程直到达到停止训练的条件。

关于具体的关于权重的偏导计算如下: 考虑特定隐藏层 $m-1$ 和下一层 m。我们用 $w_{i,j}^{m}$ 来表示 $m-1$ 层的第 j 个神经元连接到 m 层第 i 个神经元的权重，y_i^m 表示 m 层激励函数前的输入，a_i^m 来表示 m 层的第 i 个神经元的输出，f 为 m 层的激励函数。按照前面的定义重新表述为

$$\begin{aligned}a_i^m &= f(y_i^m)\\ y_i^m &= \sum_j w_{i,j}^m a_j^{m-1}\end{aligned} \tag{2-33}$$

则有每层权重的偏导计算公式为

$$\frac{\partial E}{\partial w_{i,j}^{m}} = \frac{\partial E}{\partial y_{i}^{m}} a_{j}^{m} \tag{2-34}$$

$$\frac{\partial E}{\partial y_{i}^{m}} = f'(y_{i}^{m}) \sum_{k} w_{k,i}^{m+1} \frac{\partial E}{\partial y_{k}^{m+1}} \tag{2-35}$$

标准的梯度下降法会累积全部训练样本的梯度再作权重的调整，也称为批量梯度下降（batch gradient descent)。这样在非常大的训练数据集上会因为轮数多，每轮梯度求和的计算量也很大使得计算的开销变得难以接受。而在随机梯度下降 (Stochastic Gradient Descent, SGD) 方法中，我们使用每个样本的梯度 $\nabla E(\boldsymbol{w}_n)$ 来近似真实梯度 $\nabla E(\boldsymbol{w})$，而在每个训练样本中都要更新权重。这样可以使得迭代的轮数大大减少，训练的效率获得很大的提升。在实际应用中，我们通常会将训练数据随机打乱，然后选取一个个小批量 (mini batch) 样本集完成每一次迭代的梯度计算和更新操作。

一个成熟的语音识别系统可能会需要成千上万小时的语音数据来训练，再考虑到深度神经网络的规模，在单机 CPU 上训练一个可实用的深度神经网络模型基本上可以算是不可能完成的任务。这需要我们通过一些方法加速训练过程。

从硬件架构的方向来考虑，大家通常都会采用一般用途的图形处理单元 (General Purpose Graphical Processing Unit, GPGPU) 加快运算速度。一块普通的家用显卡即可带来相比于顶级桌面 CPU 超过十倍以上的速度提升。但这并不能完全满足超大规模语料训练的需求，尤其是在使用更复杂的网络结构时。异步随机梯度下降 (asynchronous SGD, ASGD) 技术[31] 的出现很大程度上使计算能力的瓶颈得到了一定程度的缓解，使得超大模型的训练越来越现实。在这个架构中，深度神经网络模型分别别存在不同的计算机中，称为参数服务器池 (parameter server pool)。参数服务器作为主节点，把模型参数发送给从节点，从节点使用训练数据的子集进行梯度计算然后再将梯度传回给主节点。异步随机梯度下降和传统随机梯度下降不同之处在于参数服务器更新参数时并不会同步加锁更新，而是分别在不同的线程各自异步更新模型。这种方法可以大幅度的减少从节点等待其他从节点完成梯度计算的时间，从而达到加快并行速度的效果。这种方法乍看上去可能会给网络的收敛带来很大的问题，但是 [32] 证明了 ASGD 仍然可以具有很好的收敛效果，实际生产环境中的应用[33] 也佐证了这一点。

2.3.3　使用神经网络进行特征学习

一种最简单的在传统 HMM-GMM 系统中应用神经网络的方法就是使用神经网路进行特征学习。这样的方法不用修改已有的语音识别框架，可以在不大改系统的基础上提高系统的性能。因此在一段时间内特征学习方法在很多系统中得到了应用。特征学习存在两种比较常见的思路：第一种是瓶颈 (bottlenec, BN) 特征。我们需要构造一个瓶颈形状的神经网络，即其中有一个隐藏层的维度比其他的隐藏层的维度相对小很多。接下来，我们既可以使用自动编码器 (auto encoder) 对网络进行无监督训练[34]，也可以令网络的输出目标为状态后验概率，通过 BP 算法进行有监督训练。训练完成后，将瓶颈后面的网络结构删去，取此时网络的输出为特征。这样获得的 BN 特征可以被认为是一种非线性的特征变换和降维技术[35]。在构建 HMM-GMM 声学模型时，我们通常将 BN 特征和传统短时特征如 MFCC 等拼接在一起，共同作为 HMM-GMM 模型的输入进行学习。[36] 工作中使用经过预训练的深度神经网络替代传统 BN 特征中常常使用的浅层网络，结合区分性训练的方法使系统的性能得到了大幅度的提升。

另一种特征学习方法为使用串联 (tandem) 特征。在 [7] 工作中，串联特征首先使用神经网络分类器估算音素的后验概率，然后将网络输出的向量通过 PCA 做正交化作为 HMM-GMM 系统输入的特征。这样的串联方法比直接使用神经网络混合模型和标准 GMM 模型的效果都要好。而 Sivadas 等人[37] 在串联特征中使用了层次化的结构，将原来单一的神经网络替换为多个神经网络，分别被训练为具有不同的功能而又层次化地组织在一起。这种方法比原有单一神经网络的参数规模少，训练时间更短，同时获得了更好的性能。

2.3.4　HMM-DNN 混合模型

随着深度神经网络在语音识别中的作用被一步步更深地挖掘，直接采用 HMM-DNN 混合模型便成了更好的选择。在 HMM-DNN 混合模型中，我们将不同状态使用的多个 GMM 模型通过一个深度神经网络代替。我们需要训练一个深度神经网络，训练目标是估算输入的语音帧在每一个 HMM 状态下的后验概率，即 $P(q_t = s|\boldsymbol{x}_t)$。为了能够正确的估算在不同状态的后验概率，我们通常需要先通过已有的 HMM-GMM 模型和标注生成训练语料的强制对齐信息 (force alignment) 作为网络训练的目标。而强制对齐信息的好坏也很大程度上影响训练好的 HMM-DNN 混合模型系统性能，[38] 的工作中人们通过迭代使用新训练好的 HMM-DNN 混合模型生成对齐信息重新训练 HMM-DNN 混合模型的方式进一步提高了系统的性能。另外，我们通常会使用相邻的多个帧的特征复合而成的特征作为神经网络的

输入，增强网络对相邻信息的利用能力。

在 HMM-DNN 混合模型中，为了对比不同网络的本身分类的性能，我们引入帧级准确率 (frame level accuracy) 这一评价指标，统计网络输出的最大后验概率对应状态相对于标注状态的准确率。

在解码时，我们需要计算 HMM 模型的似然 $P(\boldsymbol{x}_t|q_t)$，有

$$P(\boldsymbol{x}_t|q_t) = \frac{P(q_t = s|\boldsymbol{x}_t)P(\boldsymbol{x}_t)}{P(s)} \tag{2-36}$$

其中，$P(s) = T_s/T$ 为从训练数据中估算的每一个状态的先验概率，T_s 为训练数据中 s 状态出现的数量，T 为状态总数。$P(\boldsymbol{x}_t)$ 只和语音数据本身相关，和状态序列无关，因此我们通常使用似然表达式为

$$\bar{P}(\boldsymbol{x}_t|q_t) = \frac{P(q_t = s|\boldsymbol{x}_t)}{P(s)} \tag{2-37}$$

[2][38] 等工作表明，使用 HMM-DNN 混合模型作为声学模型的系统在性能上大幅度超过了同期使用 HMM-GMM 作为声学模型的系统。从此，人们开始逐渐转向研究如何在这一框架下如何更好的应用神经网络，进一步提高系统性能。

2.4　本章小结

在本章中我们对语音识别系统中的基本知识进行了介绍，同时对神经网络自身已经在语音识别中的应用的历史进行了回顾。这些内容为后文中关于深度神经网络的研究奠定了基础。

第 3 章　语音识别中的稀疏神经网络

3.1　引言

在语音识别中使用深度神经网络的一个很大的优势是我们不需要对模型的结构和数据的分布有人工的假设，而是可以灵活地由数据驱动，最终得到一个更好的模型。大家在实际生产环境中会训练使用越来越大规模的神经网络模型，现在一个在实际生活中应用的用于语音识别系统声学模型的深度神经网络可以有 8 个隐藏层，每层有 2000-4000 个神经元，总计参数可达 1 亿个[39]。但是这其中却很可能有不少的参数是冗余的[40]，这不仅会在识别时带来更大的计算量负担，同时还会有过拟合的风险，降低模型的可扩展性。

解决过拟合问题的一种常见方式是在目标函数中加入正则化约束，称为正则化方法 (Regularization)。在模型训练时使用基于范数的正则化方法，可以倾向于产生一个稀疏的模型，提高模型的可扩展性。我们还可以直接移除掉一些神经网络中的神经元之间不重要的连接，提高网络的泛化能力。而最近流行的 Rectifier 激励函数和 Dropout 方法等，则是从改变网络结构或激励函数的角度入手，最终使得网络的权重或输出达到了稀疏的效果。这些方法都可以被认为是神经网络中的稀疏化方法。

生理学和神经科学研究表明，人脑在表达或接受信息时会分布式地激发一部分神经元[41]，而且比例小到只有 1% ~ 4%[42]。而这与我们现在平时训练出来的神经网络结构相差甚远。因此，我们认为，在深度神经网络中应用稀疏化方法，相当于引入了神经元工作方式这一先验知识，可以有助于改善模型的过拟合问题，提高模型的可扩展性，同时还可以减低模型的计算量，使复杂模型在实际生活中更加可用。本章主要讨论各种稀疏神经网络在语音识别中的应用。首先我们介绍在语音识别中应用于神经网络上的稀疏化方法，然后提出一种依据对角化假设 Hessian 矩阵的深度神经网络裁剪方法。通过对比我们的裁剪方法与基于权重大小的裁剪方法的训练效果和语音系统识别率，我们证实该方法可以在相同的稀疏度下比另一种方法获得更好的性能。

3.2　神经网络中的稀疏化方法

3.2.1　目标函数正则化方法

正则化方法是一种广泛应用于统计学和机器学习领域的模型训练方法，用于减少模型的过拟合问题。该方法通过惩罚过大的模型参数的方式来达到这一效果，在原误差函数的基础上增加正则项。

和稀疏化最直接相关的正则化方法是使用 ℓ_0 范数

$$R_0(\boldsymbol{w}) = \|\boldsymbol{w}\|_0 = \#\{k = 1 \cdots K | w_k \neq 0\} \tag{3-1}$$

其中 K 为神经网路中权重的总数。

ℓ_0 范数直接反映了非零权重的个数，但是对于它的优化问题是 NP-完全的，无法在实际问题中使用。所以我们通常会使用其他的一些范数来近似 ℓ_0 范数，如最常使用的两种范数为 ℓ_1 范数

$$R_1(\boldsymbol{w}) = \|\boldsymbol{w}\|_1 = \sum_k^K |w_k| \tag{3-2}$$

和 ℓ_2 范数

$$R_2(\boldsymbol{w}) = \|\boldsymbol{w}\|_2 = \sum_k^K w_k^2 \tag{3-3}$$

在加入了正则项约束后，训练的目标函数可以改写成

$$E_R(\boldsymbol{w}) = E(\boldsymbol{w}) + \lambda R(\boldsymbol{w}) \tag{3-4}$$

其中 $E(\boldsymbol{w})$ 为原始的均方误差或交叉熵误差函数，λ 为线性插值的权重，也被称为正则化权重。

ℓ_2 正则化方法又称权重衰减 (weight decay)，被证明可以有效的减少过拟合[43]。而 ℓ_1 更倾向于产生权重稀疏的结果，[44] 将 ℓ_1 正则化应用在编解码器的瓶颈 (bottleneck) 层上取得了不错的效果。

3.2.2　Rectifier 激励函数

在人工神经网络中最常使用的激励函数是 ‘S’ 型 sigmoid 函数或 tanh 函数，如图3.1。而在生物学对神经元的研究中，神经元实际的激励函数存在一个触发的过程，如 LIF 函数 (leaky integrate-and-fire)[45]。这些差异驱使人们尝试在人工神经网络中更好的模拟生物神经元的工作方式。在工作 [46][47][48] 中，人们尝试在深度

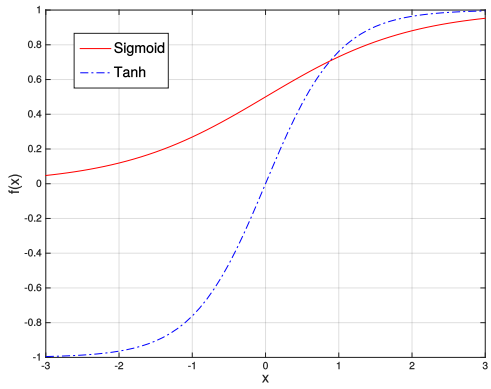

图 3.1　常用人工神经网络激励函数

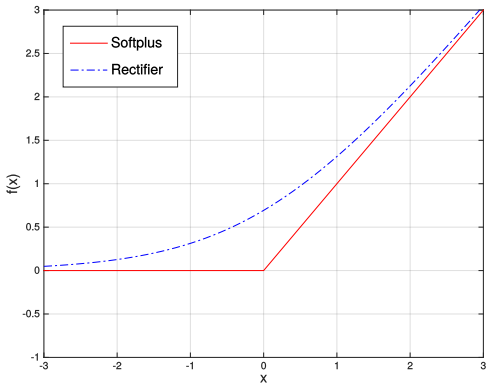

图 3.2　Rectifier 激励函数

神经网络中使用 Rectifier 函数作为激励函数，定义为

$$\text{rectifier}(x) = \max(0, x) \tag{3-5}$$

使用这样的激励函数具有以下几个优点：

- 使用 Rectifier 激励函数可以使得神经网络较容易地获得一个稀疏的表现形式。仅将激励函数替换为 Rectifier，我们就可以神经网络中 50% 的神经元的输出为 0。配合使用正则化方法可以进一步提高这个比例。
- 使用 Rectifier 激励函数相当于只是对原先神经元输出的一个线性选择，使得训练时梯度的计算更容易，同时也减少了前向传递时的计算量。

也有人认为 Rectifier 函数在 0 处的不可导可能使梯度的传递发生问题，于是提出了另一种平滑版本的 softplus 函数，期望训练可以变得更容易。softplus 函数定义为[49]

$$\text{softplus}(x) = \log(1 + e^x) \tag{3-6}$$

3.2.3　Dropout 方法

Dropout 方法也是一种为解决过拟合问题而提出来的神经网络训练方法。它的核心思路是在在网络训练的时候以概率 p 随机的丢掉神经网络中的一部分神经元与它们相连的全部连接。假设网络的神经元个数为 n，则神经网络共有 n^2 数量级的权重，同时可以被视为包含 2^n 个共享权重的小网络，每个小网络都包含完整神经网络神经元的一个子集。作者认为[50]，这样随机的丢弃过程每次都相当于从完整的神经网络中随机抽样出一个小网络，并对他进行参数调整。这样可以近似达到使用不同子集的数据训练多个不同的神经网络的效果，同时又大大地减少了计算量。而在计算网络前向传递结果时，我们又可以使用另一种近似的方法，将被随机丢掉的神经元相连的权重乘以丢弃概率 p，然后计算完整神经网络的传递

结果作为全部小网络输出结构平均的近似。实验证明，Dropout 方法可以有效地减少泛化错误，提高了模型的可扩展性。

3.2.4　神经网路权重裁剪

网络权重裁剪 (network weight pruning) 直接通过某种特定的准则显式地裁剪掉冗余或不重要的权重，只留下必须的结构。这种方法相比于前面的稀疏化方法的一个显著优点是可以方便地控制模型参数的规模，减小模型占用的储存空间，而且可以减少网络前向传递过程中的计算量，提高运算效率。我们的工作主要集中在网络权重裁剪这种稀疏化方法上。

网络权重裁剪在历史上曾经被很多研究者所采用[51][52][53][54]。Dong Yu 等人[40] 提出了在语音识别中对声学模型所使用的深度神经网络进行权重裁剪的方法。他们的方法以权重自身的大小即 $|w_{i,j}|$ 为裁剪的准则，将神经网络中权重值较小的连接裁剪掉。这样的网络裁剪获得了很好的效果：即使裁剪掉 90% 的权重，语音识别的识别率也能很大程度上不损失太多。

我们的工作和 Dong Yu 的工作相关，都应用了网络权重裁剪这一稀疏化方法。不一样的地方在于我们使用了不同的裁剪准则。[40] 工作中的裁剪准则只和权重大小相关，认为权重值小即表示权重不重要。这不是一种非常合理的假设。而我们试图从最小化误差函数改变的角度出发，利用误差函数的二阶信息指导裁剪过程，从理论上可以获得更好的效果。

误差函数的二阶信息可以通过 Hessian 矩阵来表示。在网络权重裁剪中应用 Hessian 矩阵信息曾被很多研究者尝试过，但考虑到当今的深度神经网络的隐层规模，完整的 Hessian 矩阵计算开销过于巨大，无法被直接应用。于是，研究者们提出了很多种 Hessian 矩阵的近似方法，比如对角化假设的 Optimal Brain Damage (OBD) 方法[52] 和向量外积假设的 Optimal Brain Surgery (OBS）方法。考虑到网络的规模问题，我们的方法将 OBD 方法应用于语音识别中的深度神经网络权重裁剪中，主要贡献是探索 Hessian 矩阵对角化假设在当今大规模的深度神经网络中是否可用这一问题。

3.3　OBD 深度神经网络权重裁剪

3.3.1　裁剪过程

我们把全部权重组成的向量记为 $\boldsymbol{w}$，维度为 K。由于权重的变化 $\delta\boldsymbol{w}$ 导致误差函数 E 的变化可以表示为

$$\Delta E = E(\boldsymbol{w} + \Delta\boldsymbol{w}) - E(\boldsymbol{w}) \tag{3-7}$$

对上式中的 $E(\boldsymbol{w} + \Delta\boldsymbol{w})$ 作 Taylor 展开，可以得到

$$\Delta E \approx \Delta\boldsymbol{w}\frac{\partial E}{\partial \boldsymbol{w}} + \frac{1}{2}\Delta\boldsymbol{w}^T H \Delta\boldsymbol{w} \tag{3-8}$$

其中 H 即为 Hessian 矩阵，由误差函数对权重的二阶偏导组成，表示为

$$H = \begin{bmatrix} \dfrac{\partial E^2}{\partial w_1 \partial w_1} & \dfrac{\partial E^2}{\partial w_1 \partial w_2} & \cdots & \dfrac{\partial E^2}{\partial w_1 \partial w_K} \\ \dfrac{\partial E^2}{\partial w_2 \partial w_1} & \dfrac{\partial E^2}{\partial w_2 \partial w_2} & \cdots & \dfrac{\partial E^2}{\partial w_2 \partial w_K} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial E^2}{\partial w_K \partial w_1} & \dfrac{\partial E^2}{\partial w_K \partial w_2} & \cdots & \dfrac{\partial E^2}{\partial w_K \partial w_K} \end{bmatrix} \tag{3-9}$$

对于一个已经训练好的网络，我们认为已经处于极值点，即 $\dfrac{\partial E}{\partial w} = 0$。此时由于权重变化导致误差函数的变化主要由二阶项构成。如果我们忽略掉 Hessian 矩阵的所有非对角元素，即假设裁剪连接之间是相互独立的，则有

$$\Delta E \approx \frac{1}{2}\sum_{i=1}^{K} h_{i,i} \Delta {w_i}^2 \tag{3-10}$$

其中 $h_{i,i} = \frac{\partial E^2}{\partial {w_i}^2}$ 为误差函数对每个权重自身的二阶偏导。由于将一个连接裁剪掉等价于将它的权重至零，所以裁剪掉某个权重 w_i 对误差函数的改变为

$$\Delta E_i \approx \frac{1}{2} h_{i,i} {w_i}^2 \tag{3-11}$$

依据 OBD 准则对深度神经网络进行权重裁剪的流程如下：

1. 按照正常流程训练好一个深度神经网络，使其收敛。
2. 对所有的训练数据，为所有的权重 w_i 计算 ΔE_i。
3. 依据计算好的 ΔE_i 按照大小顺序对网络进行裁剪。
4. 固定已经被裁剪的权重值不变，对裁剪后的网络重新训练使其收敛。

上面的步骤中，第二步的计算可以通过一轮反向误差传播算法完成对所有权重的变化量计算，花费略多于一轮正常训练的时间。由于裁剪后的网络已经相对

完善，最后一步中的重新训练过程只需要在低学习率下微调即可，不会进行太多次迭代。整个权重裁剪流程相对正常训练的额外时间开销并不会很大。

3.3.2　Hessian 矩阵的反向误差传递

在此节中我们描述如何计算 Hessian 矩阵中的元素 $h_{i,i}$。这里我们使用和2.3.2小节中一样的表述形式。考虑到一条神经元之间的连接 (m, i, j)，对应的 Hessian 矩阵元素 $h_{k,k}$ 可以表示为

$$h_{k,k} = \frac{\partial^2 E}{\partial w_{i,j}^{m\,2}} = \frac{\partial^2 E}{\partial (y_i^m)^2}(a_j^m)^2 \tag{3-12}$$

那么，误差函数关于 y_i^m 的二阶导数可以由下式求得

$$\frac{\partial^2 E}{\partial (y_i^m)^2} = f'(y_i^m)^2 \sum_l w_{l,i}^2 \frac{\partial^2 E}{\partial (y_l^{m+1})^2} + f''(y_i^m)\frac{\partial E}{\partial a_i^m}. \tag{3-13}$$

其中，我们需要使用到传统训练反向误差传递中的一阶传递结果。对于误差函数为交叉熵，输出层的归一化函数为 softmax 的情况，可以得到反向误差传递的初始传递值为

$$\frac{\partial E}{\partial (y_i^M)} = \sum_n \{a_i^M(n) - t_i(n)\} \tag{3-14}$$

$$\frac{\partial^2 E}{\partial (y_i^M)^2} = \sum_n a_i^M(n)(1 - a_i^M(n)) \tag{3-15}$$

3.3.3　实际应用

在实际应用中，我们基本是按照上面的流程来做的。除了有一些地方作了额外的修正，包括

- 计算出来的 Hessian 矩阵中的元素 $h_{k,k}$ 有出现负值的情况。这与进行裁剪的网络已经收敛的假设相违背。我们认为这与网络的复杂性以及对角化假设有关。在实验中我们忽略掉了这些奇异的权重。

3.4　实验环境说明

3.4.1　数据集

实验在一个中文的连续语音识别数据库上进行。该数据库由腾讯公司所提供，绝大部分为在线应用收集的录音，说话人的分布范围很广，麦克风和背景环境也

多种多样，和日常生活使用场景非常贴近。我们从中选择了 100 个小时的语料作为训练数据，7000 句语音作为开发数据，还有 17 个小时共 27000 句语音作为测试数据。

3.4.2　实验设置

整套训练环境以 Kaldi[55] 工具为基础进行搭建。Kaldi 是当前非常流行的一个开源语音识别工具包，每天都有全世界范围内的大量语音识别研究人员向其提交更新。我们使用的 HMM-DNN 系统基线系统以 Kaldi 中 Karel Vesely 的深度神经网络实现为基础所构建。

为深度神经网络提供对齐训练标注的基础的 HMM-GMM 系统包含 60 个音素和 3580 个共享上下文相关音素子状态。基线 HMM-DNN 系统使用的声学特征为 40 维 Fbank 特征，深度神经网络的前端处理工作包括以句子为单位的倒谱均值归一化 (Cepstral Mean Normalization, CMN)，叠加前五帧后五帧总计十一帧作为上下文输入，做线性判别分析 (Linear Discriminant Analysis, LDA) 降维至 200 维。词表中包含 11 万个中文词，语言模型为一个基于词的三元语言模型。我们选取深度神经网络的规模为四个隐藏层，每个隐藏层 1200 个神经元。隐藏层的激励函数选择 sigmoid 函数。输出层为 3580 维，和共享上下文相关音素子状态一一对应。训练方法选择随机梯度下降法，批量大小设为 256 帧。学习率的调整策略和标准流程中相同，通过在开发集上的误差函数的值的变化来折半减少学习率，直至开发集上的帧准确率提高小于 0.1。正常训练中的初始学习率设为 0.008，裁剪后重新训练的学习率设为 0.001。

3.5　实验结果与分析

首先我们研究在裁剪前后深度神经网络关于训练目标函数的变化情况。我们对比以权重大小为准则的网络裁剪方法 (记为 Magnitude) 和我们使用对角化 Hessian 矩阵为准则的网络裁剪方法 (记为 OBD) 在作重训练之前/重训练之后帧准确率的差别。我们按照固定的阈值设定来对 Magnitude 方法中的权重值大小或 OBD 方法中的 ΔE_i 大小进行裁剪，通过不同的阈值获得不同的稀疏度。使用 OBD 方法作网络裁剪得到的网络每一层详细的稀疏情况如表3.1。

稀疏度					
整体情况	第一层	第二层	第三层	第四层	第五层
0.50	0.76	0.55	0.58	0.48	0.44
0.25	0.57	0.25	0.32	0.26	0.21
0.12	0.42	0.08	0.14	0.15	0.10
0.06	0.32	0.04	0.07	0.10	0.06

表 3.1　使用 OBD 方法裁剪后的网络的稀疏情况

表3.2，3.3分别给出了两种方法在训练集和开发集上的帧准确率的对比。

	帧准确率%			
	Magnitude		OBD	
稀疏度	重训练前	重训练后	重训练前	重训练后
1.00	47.63	-	47.63	-
0.50	45.02	46.70	46.55	47.43
0.25	33.87	46.83	41.13	47.06
0.12	12.84	44.66	27.41	45.34
0.06	1.31	41.81	14.43	42.92

表 3.2　训练集合上两种方法的帧准确率对比

	帧准确率%			
	Magnitude		OBD	
稀疏度	重训练前	重训练后	重训练前	重训练后
1.00	45.32	-	45.32	-
0.50	43.88	45.32	44.79	45.29
0.25	34.81	45.29	41.06	45.24
0.12	13.63	44.60	28.56	44.96
0.06	1.25	42.66	15.34	43.80

表 3.3　开发集合上两种方法的帧准确率对比

从结果中我们可以发现，在训练集上使用 OBD 方法做裁剪比使用权重大小做权剪效果要好，特别是当裁剪比例很大的时候。而在开发集上，使用 OBD 方法做裁剪虽然在稀疏度比较大的时候没有体现出优势，但是在稀疏度比较低的时候还是可以获得更好的结果。另一方面，我们可以看到重新训练这个过程对于两种网络权重裁剪方法都非常重要，由于网络裁剪造成的准确率的损失可以很大程度通过重新训练来恢复，缩小两种权重裁剪方法的差距。但是，在稀疏度不太高的时候使用 OBD 方法做网络裁剪总是可以获得比按权重大小来做裁剪更好的效果。

接下来我们考察两种裁剪方法在完整语音识别系统中的表现。我们使用词错误率来衡量系统的性能。实验结果如表3.4所示。

	词错误率%	
稀疏度	Magnitude	OBD
1.00	24.04	
0.50	24.01	24.14
0.25	24.36	24.32
0.12	25.41	25.09
0.06	27.64	26.49

表 3.4　语音识别系统的词错误率比较

我们可以看到，即使我们将网络裁剪到稀疏度非常低 (~ 6%)，语音识别系统的功能也可以很大程度上得到保留。对于极度稀疏的情况，使用 OBD 进行网络裁剪能够获得比以权重大小为基础的裁剪方法更好的效果。实验结果表明，在轻微裁剪的情况下，网络仍然遗留了足够大的参数空间供优化方法进行参数调整，相比之下裁剪方法并不是最重要的；而在重度裁剪的情况下，只有很少的神经元被保留了下来，不恢复权重的重新训练能够恢复网络功能的机会也将大大受限。此时，一个更好的裁剪方法就显得尤为重要。使用 OBD 进行网络权重裁剪，可以更倾向于移除那些不重要的权重，使得裁剪后的网络受到的影响尽可能的减少。

3.6　本章小结

在本章中，我们首先回顾了几种常见的神经网络稀疏化方法，包括在目标函数中使用正则化约束范式，使用触发式的线性的网络激励函数，在神经网络训练中应用 Dropout 方法等。接着，关于神经网络权重裁剪这种特定的稀疏化方法，我们提出在 HMM-DNN 混合模型中依据对角化 Hessian 矩阵假设对 DNN 进行权重裁剪的方法。我们把这种方法和前人工作中直接依据权重大小进行权重裁剪方法的进行对比实验，证明了该方法在相同的裁剪比例下可以获得更好的效果。后续的工作包括如下几个方面：

1. 尝试探索神经元裁剪。相比于权重裁剪，神经元裁剪可以使得权重矩阵的分布更为集中，有利于提高计算速度。
2. 结合权重裁剪方法与其他稀疏化方法。权重裁剪方法与其他稀疏化方法并不冲突，可以同时使用。参考同时使用线性激励函数和 ℓ_1 正则化方法可以进一步增强模型的可扩展性，稀疏化方法的协同使用可能会取得更好的效果。

3. 探索快速稀疏矩阵乘法算法。考虑到桌面 CPU 中单指令流多数据流 (Single Instruction Multiple Data, SIMD) 控制器的大范围应用，稀疏矩阵乘法无法取得完全和稀疏度成正比的速度加成。通过修改裁剪方法或对稀疏矩阵进行变换，结合高效的稀疏矩阵乘法实现，裁剪后的神经网络将变得更加实用。

第 4 章　使用隐藏知识训练时间递归神经网络

4.1　引言

时间递归神经网络是一类神经元连接包含有向环的神经网络。和前馈神经网络不同，时间递归神经网络可以维护内部存储状态，对于序列化的输入可以提供更好的函数表达能力。在手写识别领域中，使用联结时间分类误差函数 (Connectionist Temporal Classification, CTC) 的时间递归神经网络获得了当前已知最好的结果[56]。

考虑到语音天然的序列化特性，语音识别研究者一直都在关注着时间递归神经网络。虽然在很长的一段时间内人们并没有获得非常满意的结果，但最近一段时间内越来越多的人成功地将时间递归神经网络应用在语音识别中。在音素识别方面，深度长短期记忆 (Long Short Term Memory, LSTM) 时间递归神经网络获得了很好的结果[57]；Sak 等人[58] 的工作使用了多达 1900 小时的训练语料展现了 LSTM 在大规模连续语音识别方面 HMM 混合系统中的优势；Graves 等人[59] 尝试使用全新的端对端 (End to End) 结构，将原有系统中的词表、语言模型、HMM 部分完全由 LSTM 替代，可以直接输出字母，在 WSJ 数据集上获得了和 HMM-DNN 基线系统相近的性能；百度的研究者们[60] 使用了类似的端对端结构，利用普通 RNN 在上千小时的噪声环境语料下获得了不错的结果。

尽管如此，在大规模连续语音识别任务中使用时间递归神经网络仍然存在着很多问题。从我们的实验结果来看，在 AURORA4 这种语料总量不大，但是非常贴近日常使用场景的数据集上，在 HMM 混合系统中直接应用时间递归神经网络从训练的目标函数优化情况来考虑确实表现优异，但是完整语音识别系统的识别率却不尽如人意。本章中我们首先对时间递归神经网络的基础概念做一个简单的介绍，然后讨论在 HMM 混合系统中的神经网络训练目标函数与系统实际性能不一致这个问题，最后提出一种 HMM 混合系统框架下利用已有深度神经网络输出提供的隐藏知识迁移训练时间递归神经网络的方法。实验结果表明该方法可以有效地改善在大规模语音识别应用中使用时间递归神经网络的效果，提高识别率。

4.2　时间递归神经网络

4.2.1　基本概念

首先我们给出一个隐藏层的标准的时间递归神经网络定义：对于输入序列 $\boldsymbol{x} = (x_1, \cdots, x_T)$，标准 RNN 按照顺序迭代计算隐藏层输出 $\boldsymbol{h} = (h_1, \cdots, h_T)$ 和输出层输出 $\boldsymbol{y} = (y_1, \cdots, y_T)$ 方式如下：

$$\begin{aligned} h_t &= \sigma(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \\ y_t &= W_{hy}h_t + b_y \end{aligned} \tag{4-1}$$

其中，

- W 依据下标表示不同的权重矩阵，
- b 为对应的偏置向量。

从公式来看我们可以看到时间递归神经网络和前馈神经网络的区别：各个输入之间不再独立，而是会相互影响。网络可以一直保持上文的信息。为了在前馈神经网络中增加上下文的信息，我们一般会在输入中叠加前后帧的信息。这样的做法为输入引入了长时信息，但前后的窗长只能在训练之前人工依经验指定，且存在过于固定不能在计算过程中动态调整的缺点。我们认为时间递归神经网络设计更贴近语音信号本身的性质。

训练时间递归神经网络的常用方法是按照时间顺序的反向误差传播 (Back Propagation Through Time, BPTT)。该方法按照时间顺序展开一个时间递归神经网络为普通的前馈神经网络。在反向误差传播的时候，相当于按反向时间顺序在展开后的神经网络上进行误差传递。按照时间顺序展开的示意图如4.1。

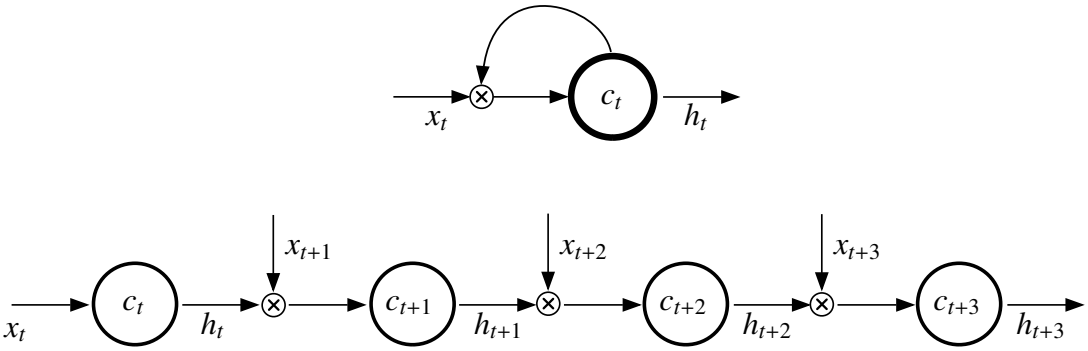

图 4.1　标准 RNN 模型及其展开结构

使用 BPTT 方法训练标准的时间递归神经网络存在梯度消失 (vanishing gradient) 和梯度爆炸 (exploding gradient) 的问题[61]。针对这个问题，Pascanu 等人[62] 提出了可以通过梯度截顶 (gradient clipping) 和限制条件来解决这两个问题。

另一方面，Hochreiter 等人[63] 提出了另一种时间递归神经网络 LSTM 模型，从网络结构角度来改善更长久范围的上下文信息利用的问题。

图4.2为一个 LSTM 神经元节点的示意图。其公式表示形式如下：

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + w_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + w_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + w_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \\
h_t &= o_t \tanh(c_t)
\end{aligned}
\tag{4-2}
$$

其中，

- σ 为 sigmoid 函数，
- i 为输入门向量，
- f 为遗忘门向量，
- o 为输出门向量，
- c 为神经元自身输出向量，
- W 同上为权重矩阵，且由神经元到门向量的权重为对角阵。

LSTM 网络在标准 RNN 结构基础上增加了输入、输出、遗忘三个门向量控制，提高了网络平衡输入和历史信息的能力，灵活的结构使其能够更好的传递梯度。

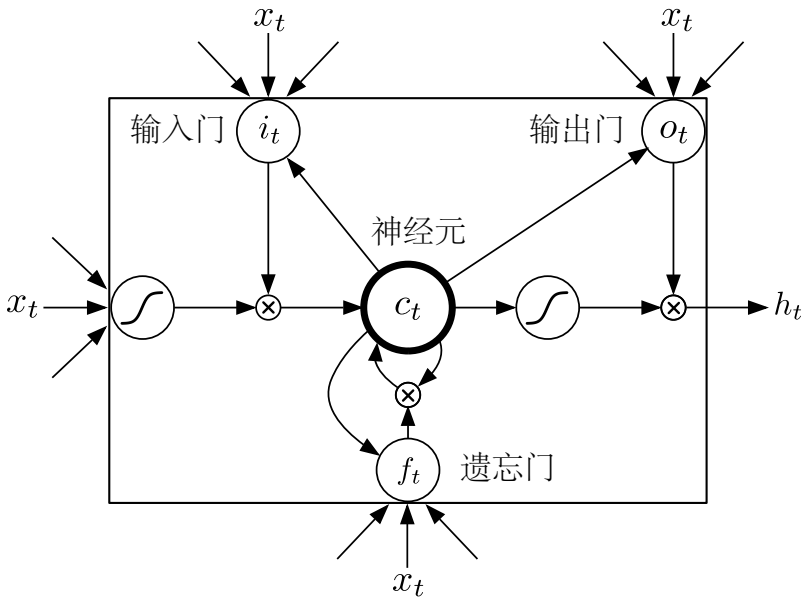

图 4.2　LSTM 神经元

为了能够利用到下文的信息，研究者还提出了双向时间递归神经网络模型 (Bidirectional RNN, BRNN)。BRNN 在利用两个单独的隐藏层从两个序列方向处理

数据，输出层为两个隐藏层输出的叠加；双向结构同样可以被应用在 LSTM 上，构成双向 LSTM 网络。考虑到深度结构在混合 HMM 语音识别系统中的成功应用，我们还可以把多个时间递归神经网络模型叠加起来，构成深度时间递归神经网络模型 (Deep RNN)。

4.2.2　语音识别中 HMM-RNN 混合模型

我们尝试使用 LSTM 构建一个使用 HMM-RNN 混合模型的语音识别系统，期望获得比基线 HMM-DNN 系统更好的性能。具体实验配置与结果在后文4.5小节中与新方法的结果一起说明。实验结果表明，从神经网络目标函数的角度来看，我们的训练成功完成，并且获得了很好的效果：无论是在训练集还是开发集上，LSTM 模型关于语料的帧准确率相比 DNN 模型都获得了极大的提升；但是与此相对的，LSTM 模型在测试集上识别率却完全没有得到提升，而是大幅下降。使用区分性训练能够在目标函数的角度更加靠近语音识别系统的实际性能，但是即使我们再次对 LSTM 模型做区分性训练，语音识别系统性能相比交叉熵训练时获得了一定提升，但仍然比 HMM-DNN 基线系统差。类似的现象在 [59][64][65] 中也提到过。Graves[59] 认为，这是由于目标函数和真实性能评价指标不一致导致的，是 HMM 混合模型自身的问题，也是端对端学习希望解决的问题。而 Chan 等人[64] 则通过在 LSTM 前后继续增加了 4 层前馈网络结构的方式才获得了比基线 HMM-DNN 系统更好的性能。

我们认为，优化交叉熵目标函数和真实性能评价指标识别率确实不一致，但是仍然可以通过一些调整使得目标函数更加靠近真实指标。在整个语音识别系统之中，音素作为语音学上的基础的单元仍然具有非常重要的意义，完全跳过音素不符合人类自己做人工语音识别过程的认知，同时会导致重要先验知识假设的缺失。因此，我们试图探寻改善时间递归神经网络模型训练效果的方法。

由于时间递归神经网络模型结构的复杂性，直接应用使用一阶信息的梯度下降方法来进行神经网络训练无法获得很好的训练效果。Martens 等人[66] 利用目标函数的二阶信息使用 Hessian-free 优化方法来训练标准结构的时间递归神经网络，并取得了不错的效果。但是该方法会造成网络训练时的计算开销增大的问题，同时我们没有发现有人可以将 Hessian-free 优化方法成功地应用于训练 LSTM 模型中。而 Sutskever 等人[67] 则使用了一种仔细设计过的冲量调整策略达到了更好的时间递归网络训练效果。然而，该方法调整参数的方法过于技巧化，无法轻易地扩展到其他情况中。

在下面的章节中，我们尝试提出一种简单但有效的训练时间递归神经网络的方法。该方法很大程度上受到了 Hinton 等人[68] 工作的启发，将深度神经网络中

的隐藏知识迁移到时间递归神经网络的训练中。

4.3　深度神经网络中隐藏的知识

在相同的数据上训练很多种不同的模型，然后将他们的输出结果做平均作为输出是一种通用的提高机器学习算法性能的方法。这种方法被称为集成方法 (ensemble method)。然而，在实际环境中使用集成方法会导致计算量增加，降低了方法的实用性。Caruana 等人[69] 的研究结果表明，我们可以把集成方法中的多个模型中的知识压缩到一个单独模型中。Hinton 等人[68] 延续了这一工作，使用了一个不同的模型压缩方法，使神经网络模型在包括语音识别在内的很多领域都获得了了令人惊讶的性能提升。

Hinton 等人认为，使用集成方法或训练好一个大规模的模型之后，可以把这其中的包含知识转移到一个小模型中。一般的误差函数的优化目标是最大化标注对应输出的概率，而对于其他的输出的概率却是平权优化的。但是，其他输出的实际概率往往会差异很大，非标注输出的概率的相对差异同样是非常重要的，某种程度上反映了模型在遇到未知输入时是如何泛化的。因此，训练好的大规模模型输出的概率是一个非常重要的信息，如果把这个信息用于训练小模型，应该会比直接使用标注训练获得更好的效果。

神经网络输出的最后一层使用 softmax 归一化函数将网络的输出转换为不同分类的后验概率。考虑 softmax 函数的一种扩展形式

$$\text{softmax}(i) = \frac{\exp(z_i/T)}{\sum_j^N \exp(z_j/T)} \tag{4-3}$$

其中，T 为温度 (temperature)，通常被指定为 1。使用一个更大的 T 值将会使 softmax 函数产生更柔和的概率分布。

进行知识转移最简单的方法就是直接使用已有模型的在一个大的 T 值下的概率分布作为新模型训练的目标。目标的 softmax 函数在训练时也要使用相同的 T 值，而在测试时使用 $T = 1$。我们称这样的训练方式为使用一个柔和目标 (soft target)。

而对于有标注的数据，同时利用数据的标注信息会获得更好的结果。我们只要使用两种不同的目标函数的线性组合就可以得到很好的结果。这两种目标函数为依据原有标注的交叉熵的误差函数和上一段中提到的使用柔和目标的交叉熵误差函数。需要注意的是使用柔和目标的交叉熵误差函数的梯度值的大小仅为原来的 $1/T^2$ 倍，所以我们在使用两种目标函数的时候要将柔和目标的目标函数的梯度值乘以 T^2。对柔和目标的误差函数使用更大的权重可以获得更好的结果。

4.4　隐藏知识迁移训练

受到上面方法的启发，我们尝试探索能否把类似的训练方法应用在时间递归神经网络中。我们认为，时间递归神经网络得益于网络结构的灵活性，相比于前馈神经网络可以对目标函数更精确地逼近。但是在数据规模不够大的情况下，只包含标注的误差函数倾向将网络优化为对所有非标注输出都为无差别的零概率结果，这反而极大地伤害了网络的可扩展能力。此时，如果可以借助已经训练好的网络的信息，将有助于提高网络的泛化能力，获得更好的结果。但是，在上节的应用场景中，我们是将集成方法模型或一个精心训练的大规模模型的信息提炼到一个小模型中。而在当前应用场景下，我们尝试把一个中等规模的深度前馈神经网络的信息转移到一个内部结构更为复杂的时间递归神经网络中。本章的主要贡献是验证相对较弱的信息通过这种信息提炼方法对于语音识别场景下 HMM 混合模型中的时间递归神经网络的训练是否有帮助。

我们通过修改训练的目标函数的方式完成隐藏知识的迁移。新的目标函数为

$$\begin{aligned} E(\theta) &= \alpha E_H(\theta) + E_S(\theta) \\ &= \sum_i^N \sum_j^M (\alpha t_{ij} + p_{ij}) \ln\{y_{ij}(\theta)\} \end{aligned} \tag{4-4}$$

其中，$E_H(\theta)$ 为原始的直接使用标注的误差函数，$E_S(\theta)$ 为使用隐藏知识的柔和目标的误差函数，α 为线性插值常量，在本文的工作中取 0.5。

新的目标函数可以被看作在原有的训练目标中加入了一个正则化的约束条件，从而可以达到使目标函数更加平滑的效果。因此，该目标函数相对原始目标函数更加易于优化，同时可以起到增强时间递归神经网络模型泛化能力的效果。

4.5　实验结果与分析

4.5.1　数据库

在本章中我们使用 AURORA4 数据库[①]。AURORA4 是由欧洲通信标准协会 (European Telecommunications Standards Institute, ETSI) 的 Aurora 工作组提供的一个用于评估大规模连续语音识别的数据集。这个数据集主要评估目标是为了考察系统在不同前端（噪声，采样率的减少，不同麦克风的信道区别，压缩，模型不匹配）环境下的鲁棒性。数据集的原始数据来自于著名的华尔街日报 (Wall Street

① http://aurora.hsnr.de/aurora-4.html

环境	麦克风	噪声类型	噪声水平
1	Sennheiser HMD-414	无	无
2	Sennheiser HMD-414	汽车噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
3	Sennheiser HMD-414	背景说话噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
4	Sennheiser HMD-414	餐厅噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
5	Sennheiser HMD-414	街道噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
6	Sennheiser HMD-414	机场噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
7	Sennheiser HMD-414	火车噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
8	18 种麦克风随机采样	无	无
9	18 种麦克风随机采样	汽车噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
10	18 种麦克风随机采样	背景说话噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
11	18 种麦克风随机采样	餐厅噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
12	18 种麦克风随机采样	街道噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
13	18 种麦克风随机采样	机场噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB
14	18 种麦克风随机采样	火车噪声	训练集 10 ~ 20 dB, 测试集 5 ~ 15 dB

表 4.1　AURORA4 数据集中不同环境的说明

Journal, WSJ) 数据库[①] 。

AURORA 数据库由训练，开发和测试集构成。Aurora 工作组把词表的范围控制在 5000 个英文单词之中，且在测试集中没有出现词表外的单词 (out of vocabulary words, OOVs)。这样做的目的是让使用数据集的研究人员尽量把目标集中在优化声学模型上，减少语言模型的影响。训练集是由来自 83 个说话人，总共 7138 句话，共 14 小时的语音数据构成。训练集有完全纯净和包含麦克风、噪声类型、噪声水平各异的 14 种不同环境情况两种选择，总量相同。具体的环境情况说明见表4.1。噪声是通过数字方式人工增加到语音中的。开发集是由不同环境的 330×14 句语料组成的 “330 子集”，而测试数据则由以完整的 NIST 92 评估集为基础，录制不同环境下的 330×14 句语料组成。为了模拟真实环境下的语音数据，我们选择包含各种环境情况的训练数据作为训练集，全部 14 种环境的测试数据作为测试集。我们认为 AURORA4 数据库可以很好地评估一个语音识别系统的声学模型在真实环境下的建模能力。

4.5.2　实验设置

训练环境同时使用 Kaldi 搭建。为网络训练提供对齐训练标注的基础的 HMM-GMM 系统包含 351 个音素和 2008 个共享上下文相关音素子状态。

① WSJ 数据库是一个由高质量朗读录音组成的大规模连续语音识别数据库。当前主流的语音识别系统在 WSJ 数据库上一般都可以获得很好的性能。但是，随着背景噪声的增加，系统的性能往往会急剧下降。

对于基线 HMM-DNN 混合模型系统，我们使用的声学特征为 40 维 Fbank 特征，深度神经网络的前端处理工作包括以句子为单位的倒谱均值归一化，叠加前五帧后五帧总计十一帧作为上下文输入，做线性判别分析降维至 200 维。语言模型使用标准的 kaldi AURORA4 数据库流程中的三元语言模型。深度神经网络的规模为四个隐藏层，每个隐藏层 1024 或 2048 个神经元。隐藏层的激励函数选择 sigmoid 函数。输出层为 2008 维，和共享上下文相关音素子状态一一对应。训练方法选择随机梯度下降法，批量大小设为 256 帧。学习率的调整策略和标准流程中相同，通过在开发集上的误差函数的值的变化来折半减少学习率，直至开发集上的帧准确率提高小于 0.1，初始学习率设为 0.008。

对于 HMM-RNN 混合模型系统，我们使用的声学特征同样为 40 维 Fbank 特征，前端处理工作只包括以句子为单位的倒谱均值归一化和 5 帧前移 (frame shifting)。语言模型与之前相同，网络结构只包含一层单向投影 LSTM 隐藏层，包含 800 个神经元，投影至 512 维输出。输出层情况与之前相同。训练方法为 [58] 中使用的按语句并行的 BPTT 方法，在一个梯度下降步骤中同时累积计算 4 句语料的梯度。设置梯度下降的冲量值为 0.9，使用和前文中类似的学习率调整策略，初始学习率 0.0001，折半系数 0.7。

4.5.3　实验结果

表4.2给出了在不同训练方法的情况下的语音识别系统中使用深度神经网络和时间递归神经网络的性能对比。

网络类型	模型大小/M	训练集 FA%	开发集 FA%	测试集 WER%
DNN/1024	21M	59.38	45.00	11.70
DNN/2048	66M	63.07	45.18	11.40
LSTM	13M	**69.53**	52.17	13.57
DNN/2048, 迁移训练	66M	61.70	47.02	11.13
LSTM, 迁移训练	13M	67.48	**53.70**	**10.84**

表 4.2　语音识别系统中不同训练方法和网络结构的性能

FA% 为帧准确率，WER% 为词错误率

首先我们关注在未使用迁移训练方法时的情况。我们可以看到，使用 LSTM 作为混合 HMM 模型中的分类器从训练目标的角度来考虑可以获得非常好的结果：相比与基线 HMM-DNN 混合模型系统，训练集和开发集的帧准确率在 HMM-LSTM 混合模型系统中都可以得到大幅度的提升。按照在基线 HMM-DNN 混合模型系统中增大模型规模带来的目标函数优化对系统性能提升的贡献程度推算，使用 HMM-LSTM 混合模型理应使语音识别系统在测试集中的实际性能得到

很大比例的提升。而结果却恰恰相反，在语音识别系统中应用标准的 LSTM 模型会导致系统性能大幅下降，甚至低于小规模的 DNN 模型。这和常见的过拟合情况并不相同：开发集是完全不参与优化目标函数过程的与训练集相独立的数据集，从训练目标函数的角度来看模型在开发集上的分类性能也得到了大幅度的提升。

基于上文的分析，我们尝试在训练 LSTM 网络时使用迁移训练的方法。迁移训练采用线性插值柔和目标的目标函数和真实标注的目标函数的做法，比例设为 2。柔和目标通过基线 2048 神经元的 DNN 获得。训练的 LSTM 和 DNN 网络和基线系统中的配置相同。从实验结果中我们可以看出，同等规模的网络在迁移训练后降低了训练集的帧准确率，但是提高了开发集的帧准确率，说明迁移训练方法可以起到进一步提升神经网络泛化能力的效果。在测试集上的词错误率结果验证了这一点。

另一方面，通过对比对前馈神经网络和时间递归神经网络在使用迁移训练和未使用时的性能对比，我们可以发现，迁移训练可以大幅度改善时间递归神经网络在完整语音识别系统中的性能，在使用相同规模的网络的情况下降低词错误率 20% 以上。在使用相同的柔和目标的情况下，模型参数更少的 LSTM 模型展现了更好的分类能力与泛化能力。我们认为，时间递归神经网络的逼近目标的能力要大大强于前馈神经网络，因此在训练时若只使用真实标注来训练，可能会使得网络的映射函数过于陡峭，缺少区分唯一标注外其他输出单元的分类能力。这样会导致虽然对于大多数帧网络的输出和真实标注相匹配，但是在不能匹配标注的帧上与实际的后验概率分布相差更大，从而降低了语音识别系统的性能。而使用迁移训练，能够利用到前馈网络学习到的后验概率分布信息，使得网络训练的映射函数更加平滑，进而获得了更好的效果。

4.6　本章小结

本章中我们首先介绍了时间递归神经网络的相关知识，然后描述了直接在语音识别中应用 HMM-RNN 混合模型带来的实际性能不匹配问题。接下来，我们分析了问题的成因，并提出一种利用已有深度神经网络输出作为柔和目标提供隐藏知识迁移训练时间递归神经网络的方法。实验结果表明该方法可以显著改善语音识别中 HMM-LSTM 混合模型的性能，并且使得单层单向小规模 LSTM 模型即可在识别率上超过大规模 DNN 模型。后续工作包括以下几个方面：

1. 探索新型系统结构。时间递归神经网络具有很强的时间序列建模能力，甚至可以直接替代现有的 HMM 模型。现在端到端的系统架构尝试也证明了这一点。我们可以利用时间递归神经网络尝试其他的系统结构，可能会获得更好

的效果。

2. 快速时间递归神经网络计算。虽然时间递归神经网络参数紧凑，但由于需要逐帧传递计算，计算时间远远长于普通深度前馈神经网络。我们可以尝试探索快速的时间递归神经网络的计算方法，使得时间递归神经网络在日常生活中更加可用。

第 5 章　跨平台在线语音识别系统实现

5.1　引言

在上面的章节中，我们着重阐述如何更好地在语音识别的声学模型中应用各种神经网络模型，增强模型的可扩展性，提高模型的性能。而在训练好模型之后，如何利用模型快速、准确地将语音对应的文本识别出来，仍然是一个非常重要的问题。

本章描述了一种语音识别系统的实现方法。该系统的目标是为各种在线应用提供语音识别服务。系统使用 HMM 混合模型声学模型和 n-gram 语言模型，通过预先生成的静态有限状态变换器解码图进行识别。系统的使用环境可以是大型的高性能服务器，也可以是小型的嵌入式便携设备。系统需要支持在线解码 (online decoding)，即尽可能早地将部分识别结果返回给用户，而不是等完整语音结束后才开始进行识别进程。系统在各个平台都需要达到实时识别的要求，即 RT 值小于 1。对于在服务器上使用的情况，系统需要支持多路并发解码，充分利用服务器的多处理器条件；而对于在小型设备上的情况，系统需采用一定的策略，在尽可能少的牺牲性能的情况下让识别速度达到要求。本章描述的语音识别系统实现完成了上述要求。

5.2　识别模块

整个系统中最核心的部分就是识别模块。识别模块的工作流程如图5.1所示。

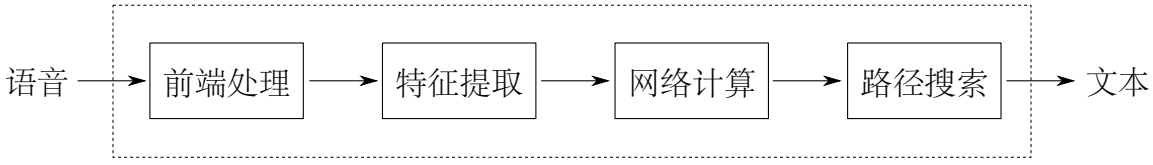

图 5.1　识别模块工作流程

语音被逐段送入到识别模块中，先经过前端处理模块做语音端点检测 (Voice Activity Detection, VAD) 等操作，过滤出需要进行识别部分的语音；接着送入特征提取模块，按与声学模型匹配的流程完成特征向量的计算；然后通过已经训练好的神经网络上计算特征向量在各个上下文相关音素子状态的概率；最后在构造好的有限状态变换器网络上搜索最优路径，并输出结果。

下面我们逐部分的详细说明我们系统在不同处理流程中的实现细节。

5.2.1　特征提取

经过前端处理后的语音首先同2.2.4小节所述的方法提取 Fbank 特征。计算 Fbank 特征的依赖的窗口长度并不大，只需要保留前一帧的语音即可。但是，在线解码对接下来的倒谱均值归一化操作影响很大。在模型训练时，为了能精确地估计倒谱均值，通常对完整的一整句语料均值的计算。但是在在线识别的任务中，为了能够尽快的获得结果，我们无法在句子完整结束后再进行均值的估计。对于这个问题，我们通常会使用在线倒谱均值归一化 (Online CMN) 来近似完整的句子的倒谱均值归一化。Online CMN 根据当前的输入的长度，计算当前部分倒谱均值；在传入新数据之后，会把当前段的倒谱均值和之前累积部分的均值做以长度为比例的线性插值。我们需要在训练和测试阶段使用相同的归一化函数。在使用 Online CMN 作为归一化函数的时候，会存在语音刚开始的一段时间内因为采样样本过少无法很好的估计倒谱均值的问题。因此，在基础的 Online CMN 基础上，我们还增加了加载和保存全局倒谱均值的功能：在还没有语音的时候，就有一个初始的全局倒谱均值作为倒谱均值的初值。默认情况下，全局倒谱均值为训练集合上的倒谱均值的平均。使用识别模块的程序还可以在合适的时候自行控制加载和保存全局倒谱均值，可以更准确的估算当前倒谱均值，以达到更好的适应用户环境的效果。

5.2.2　网络计算

我们需要在预先训练好的声学模型上计算特征向量在子状态上的概率。回顾神经网络的传递函数，我们可以发现，计算量主要集中在矩阵乘法上。对于一般的深度前馈神经网络，各个帧之间是相互独立的，我们可以把多个帧的向量拼在一起，形成一个矩阵。在计算时，用矩阵与矩阵乘法替换矩阵向量乘法，可以一并得到多个帧的网络输出。相比逐帧计算，这种做法可以更好的利用处理器中的单指令流多数据流控制器，提高运算效率。同时，对于矩阵与矩阵乘法，在现在常用的 Intel x86 处理器上，存在着大量的优化得非常好矩阵运算库，包括开源的自动优化线性代数库 Automatically Tuned Linear Algebra Software (ATLAS)[①]，以及由英特尔公司提供的商业的数学运算库 Math Kernel Library(MKL)[②] 等。因此，在当前主流的处理器上，尤其是支持高级向量扩展 (Advanced Vector Extensions,

① http://math-atlas.sourceforge.net/

② https://software.intel.com/en-us/intel-mkl

AVX) 指令集的处理器，对于一般规模的深度前馈神经网络，矩阵乘法并不是影响实时率的一个非常重要的因素。但是，对于时间递归神经网络，特别是 LSTM，由于需要逐帧计算输出，本身计算函数又复杂，使用 CPU 做运算不能获得很好的实时性保证。这个时候我们可能需要借助 GPU 来进行网络计算。

而在小型嵌入式设备上的网络计算需要进行一些特殊的处理。目前最常见的嵌入式设备上的 CPU 架构是 ARM 架构，广泛地应用在手机、平板、可穿戴设备等各种各样的小型设备中。该系列架构的片上系统 (System On Chip, SOC) 具有体积小巧、功耗低等特点。但与之相对的，在计算能力方面，特别是浮点运算的性能，ARM 架构的处理器相比于主流 x86 架构处理器仍存在不小的差距。如果将中等规模的深度前馈神经网络直接应用在小型设备上是完全无法满足实时性要求的。

对于上述问题，我们可以通过如下方法来解决。参考 [70] 中的做法，我们可以把神经网络中的权重压缩至 8 位有符号整数，在计算隐藏层输出的时候也将结果量化到只有 8 位。这样会变成使用整数矩阵乘法完成仿射函数的计算，输出层的激励函数计算采用打表法即可。这样可以在基本不损失性能的情况下既减少模型的存储空间，同时提高了模型的计算速度。另外，我们可以使用跳帧 (frame skipping)[71] 技术，在连续的 n_s 帧中，我们只计算 1 帧的网络输出，用该帧的输出来近似其他帧的网络输出。实验表明，当 $n_s = 2$ 时，我们可以在可以接受的性能的损失下提高一倍网络计算的速度。

5.2.3　路径搜索

路径搜索部分是识别模块的核心功能，直接关系到结果的输出。当前主流的解码器实现都采用了预先构造有限状态变换器解码图[72]，然后在图上进行路径搜索的方法来实现解码过程。但是，在构造有限状态变换器的完整程度上却存在着一些差别：kaldi 中的内置解码器选择将 HMM 模型也完整地构造入有限状态变换器解码图之中，形成了完整的 HCLG 有限状态变换器；而如著名的解码器实现 Juicer① 选择不将 HMM 构造进解码图中，仅生成 CLG 有限状态变换器。解码器需要在搜索的时候动态地扩展 HMM 状态。这两种方法各有优劣，分别适用于不同的应用场景：在高性能服务器端使用时，服务器的内存相对充裕，我们认为在这样的环境下使用 HCLG 的解码器更加合适。HCLG 解码器在构图时完整的包含了 HMM 模型，使得有限状态变换器的输入即为 HMM 中的一个子状态。这样会使得解码器的实现变得非常简单，经过预先优化的完整解码图也可以进一步降低解码操作的时间花销；而对于内存相对吃紧的嵌入式环境来说，占用内存更小的

① http://juicer.amiproject.org/juicer/

CLG 解码器可能是一个更合适的选择。从另外一个方面看，如果当系统需要动态的语法支持，需要动态重新生成有限状态变换器解码图的时候，组合时间更短的 CLG 解码器便可以发挥它的优势。

使用预先构造的有限状态变换器的解码器存在着一个问题：我们需要将语言模型完整地组合进最终的有限状态变换器中，但是后面组合上来的上下文信息变换器和 HMM 模型变换器会使得最终的有限状态变化器大小相比语言模型本身对应的有限状态变换器膨胀很多倍。这很大程度上限制了我们能够在有限状态变换器中使用的语言模型的大小。一个常见的解决语音模型大小限制的方法是使用重打分 (rescore) 技术[73]，使用一个小的语言模型构造有限状态变换器解码图，然后在解码器生成的网格 (lattice) 或 N-best 结果上利用更大的语言模型重新计算语言模型部分的分数，并最终得到新的最优结果。这种方法需要等到一句语音识别才能重打分，无法应用到在线场景之中。这里我们使用 Takaaki 等人[74] 提出的实时 (on-the-fly) 重打分方法。该方法在解码时动态组合一个完整的 HCLG 或 CLG 有限状态变换器和一个大的语言模型对应的有限状态变换器，并在动态生成的有限状态变换器上进行解码。

5.3　系统结构

我们将识别模块封装为动态链接库的形式，提供了几个简单的 C 语言接口供其他应用程序直接调用。在此基础上，为了方便在服务器上并发的使用语音识别功能，我们还实现了一个提供语音识别服务的服务器端程序。系统架构如图5.2所示。

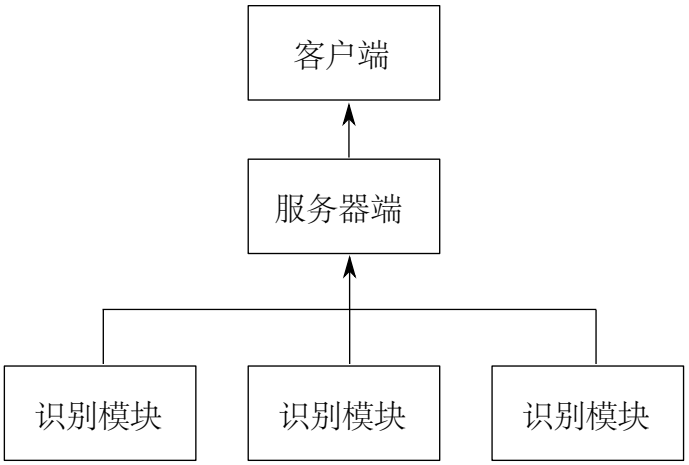

图 5.2　语音识别系统结构

我们可以方便地在各种环境中创建一个 TCP 连接与服务器端通信，把语音数

图 5.3　手机本地识别应用场景

图 5.4　在线服务应用场景

据分块发送给服务器端做识别。服务器会不定时的返回识别结果。该服务器端程序的工作流程如下：

1. 根据配置文件的设定，加载模型文件，在单独的线程启动指定数目的识别模块。
2. 监听 TCP 端口，等待客户端连接。
3. 当客户端连接成功后，按照固定的协议接收用户输入的语音；寻找一个空闲的识别模块，将语音转发进行识别。
4. 若当前系统繁忙，没有空闲的识别模块，则将语音数据存入至队列中。
5. 待识别模块处理完前一个任务后从队列中依次取出新的语音进行识别。
6. 将识别结果在同一个连接内发送给客户端。

5.4　系统演示

下面是两种在不同环境下使用本章中提到的语音识别系统演示。图5.3为在一部安卓手机上本地运行的语音识别系统演示。图5.4演示在微信公众号中使用远程服务器提供的更精确的语音识别服务。

5.5　本章小结

本章逐部分介绍了如何实现一个使用 HMM-DNN 混合模型作为声学模型的实时在线语音识别解码器。对于不同的使用场景，后续工作分别有不同的优化目标：对于服务器端，我们通常会使用一个非常大的语言模型，所以路径搜索部分成为了需要着重考虑的地方。另外，在语言模型方面使用时间递归神经网络能够获得很好的效果，所以在解码器中支持使用时间递归神经网络语言模型进行实时解码也是很重要的一部分工作。对于嵌入式端，神经网络输出计算仍然占解码时间很大的一部分比例，我们需要进一步优化嵌入式设备上的网络计算模块，并且探索更好的裁剪方法。

第 6 章　总结与展望

6.1　本文工作总结

如何在声学模型中使用深度神经网络提高系统的性能是语音识别领域的一个重要的研究内容。本文中我们从两个方向探索了 HMM 混合神经网络模型中神经网络的改进思路：

1. 稀疏神经网络

我们认为，在神经网络中应用稀疏化方法有助于增强模型的泛化能力，提高模型的鲁棒性。而网络权重裁剪这种特定的稀疏化方法同时还可以有效地减小模型大小，提高运算效率。在前人使用权重大小作为准则对网络进行裁剪的基础上，我们从最小化目标函数改变的角度尝试寻找一种更精确的网络权重裁剪准则。该方法基于对角化 Hessian 矩阵的假设，从理论上能够更精确地选择相对不重要的连接权重。实验结果证实了这一点，我们的权重裁剪方法可以获得比前人的裁剪方法更好的效果。

2. 时间递归神经网络

时间递归神经网络具有记忆长时上下文信息的能力，理论上可以更好的建模时间序列信号。但是在大规模连续语音识别系统中，直接使用时间递归神经网络虽然在训练目标函数上取得了很好的效果，但却无法等价地转化为识别率的提升，反而使系统性能下降。我们尝试探寻训练目标与真实性能不一致的原因及解决方案。受到模型压缩方法思路的启发，我们尝试在时间递归神经网络的训练目标函数中加入柔和目标，利用已有深度前馈神经网络的知识更好地优化时间递归神经网络的映射函数。结果表明这种方法可以大幅提高 HMM 混合模型中使用时间递归神经网络的效果，并最终达到了比基线深度前馈神经网络系统更好的性能。

在上述工作基础上，我们还实现了一个跨平台的语音识别解码器，可以高效地使用 HMM 混合神经网络模型进行实时在线语音识别。解码器可以应用在服务器上，支持多路并发语音识别，也可以在近似算法和模型裁剪的帮助下应用于小型嵌入式设备之中，在不损失过多性能的情况下保持实时性。

6.2　未来工作展望

我们相信，现在的语音识别系统还没有完全发挥神经网络结构的能力，更好地在语音识别系统中应用深度学习技术能够帮助语音识别技术在日常生活中更加普及化，实用化。作者认为存在以下几个可能的改进方向：

如何自主学习深层神经网路结构可能是一个重要的研究内容。现在在实际生活中使用的神经网络的规模和结构的选择往往是人工试错得到的，缺少理论支持。同时，全连接结构中的很多权重参数很可能是冗余的，影响了网络的泛化能力。我们之前使用的权重裁剪方法只是一种非常简单的简化结构的尝试，如果能够在神经网络的训练过程中自主地学习调整网络的结构，将有助于减少人工设定训练参数的盲目性，使得模型的可扩展能力得到更进一步的提升。

如何将时间递归神经网络更好地应用于语音识别之中也是一个重要的研究方向。我们使用的方法在一定程度上改善了训练目标和系统识别性能不一致的问题，但是这个问题仍然很大程度上影响了系统性能的进一步提升。使用时间递归神经网络做端到端的语音识别是一个不错的思路，在这种方法中人们使用时间递归神经网络替代掉原有系统中的隐马尔可夫模型基础框架，取得了不错的效果。我们认为，如果能够减少现有系统中人工模型的假设，但是仍然保留人类在语音学上的的一些先验知识，如音素的信息等，可能会获得更好的结果。

参考文献

[1] Huang X D, Ariki Y, Jack M A. Hidden Markov models for speech recognition, volume 2004. Edinburgh university press Edinburgh, 1990.

[2] Hinton G, Deng L, Yu D, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. Signal Processing Magazine, IEEE, 2012, 29(6):82–97.

[3] Fiscus J G, Ajot J, Garofolo J S. The rich transcription 2007 meeting recognition evaluation. Multimodal Technologies for Perception of Humans. Springer, 2008: 373–389.

[4] Haeb-Umbach R, Ney H. Linear discriminant analysis for improved large vocabulary continuous speech recognition. Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on, volume 1. IEEE, 1992. 13–16.

[5] Leggetter C J, Woodland P C. Maximum likelihood linear regression for speaker adaptation of continuous density hidden markov models. Computer Speech & Language, 1995, 9(2):171–185.

[6] Li Y, Erdogan H, Gao Y, et al. Incremental on-line feature space mllr adaptation for telephony speech recognition. INTERSPEECH, 2002.

[7] Hermansky H, Ellis D P, Sharma S. Tandem connectionist feature extraction for conventional hmm systems. Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on, volume 3. IEEE, 2000. 1635–1638.

[8] Konig Y, Heck L, Weintraub M, et al. Nonlinear discriminant feature extraction for robust text-independent speaker recognition. Proc. RLA2C, ESCA workshop on Speaker Recognition and its Commercial and Forensic Applications, 1998. 72–75.

[9] Woodland P C, Povey D. Large scale discriminative training of hidden markov models for speech recognition. Computer Speech & Language, 2002, 16(1):25–47.

[10] Povey D, Woodland P C. Minimum phone error and i-smoothing for improved discriminative training. Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on, volume 1. IEEE, 2002. I–105.

[11] Povey D, Kanevsky D, Kingsbury B, et al. Boosted mmi for model and feature-space discriminative training. Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on. IEEE, 2008. 4057–4060.

[12] Povey D, Kingsbury B, Mangu L, et al. fmpe: Discriminatively trained features for speech recognition. ICASSP (1), 2005. 961–964.

[13] Povey D, Burget L, Agarwal M, et al. Subspace gaussian mixture models for speech recognition. Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on. IEEE, 2010. 4330–4333.

[14] Jaitly N, Nguyen P, Senior A W, et al. Application of pretrained deep neural networks to large vocabulary speech recognition. INTERSPEECH. Citeseer, 2012.

[15] Bengio Y, Lamblin P, Popovici D, et al. Greedy layer-wise training of deep networks. Advances in neural information processing systems, 2007, 19:153.

[16] Yu D, Deng L, Dahl G. Roles of pre-training and fine-tuning in context-dependent dbn-hmms for real-world speech recognition. Proc. NIPS Workshop on Deep Learning and Unsupervised Feature Learning, 2010.

[17] Dahl G E, Sainath T N, Hinton G E. Improving deep neural networks for lvcsr using rectified linear units and dropout. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013.

[18] Miao Y, Metze F, Rawat S. Deep maxout networks for low-resource speech recognition. Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, 2013. 398–403.

[19] Abdel-Hamid O, Mohamed A r, Jiang H, et al. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition. Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on. IEEE, 2012. 4277–4280.

[20] Veselỳ K, Ghoshal A, Burget L, et al. Sequence-discriminative training of deep neural networks. INTERSPEECH, 2013. 2345–2349.

[21] Hermansky H. Perceptual linear predictive (plp) analysis of speech. the Journal of the Acoustical Society of America, 1990, 87(4):1738–1752.

[22] Qi J, Wang D, Xu J, et al. Bottleneck features based on gammatone frequency cepstral coefficients. INTERSPEECH, 2013.

[23] Baum L E, Petrie T. Statistical inference for probabilistic functions of finite state markov chains. The annals of mathematical statistics, 1966. 1554–1563.

[24] Dempster A P, Laird N M, Rubin D B. Maximum likelihood from incomplete data via the em algorithm. Journal of the royal statistical society. Series B (methodological), 1977. 1–38.

[25] Nádas A, Nahamoo D, Picheny M A. On a model-robust training method for speech recognition. Acoustics, Speech and Signal Processing, IEEE Transactions on, 1988, 36(9):1432–1436.

[26] Normandin Y, Lacouture R, Cardin R. Mmie training for large vocabulary continuous speech recognition. Third International Conference on Spoken Language Processing, 1994.

[27] Normandin Y, Morgera S D. An improved mmie training algorithm for speaker-independent, small vocabulary, continuous speech recognition. Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on. IEEE, 1991. 537–540.

[28] Valtchev V, Odell J, Woodland P C, et al. Mmie training of large vocabulary recognition systems. Speech Communication, 1997, 22(4):303–314.

[29] Lippmann R P. Review of neural networks for speech recognition. Neural computation, 1989, 1(1):1–38.

[30] Polyak B T. Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 1964, 4(5):1–17.

[31] Le Q V. Building high-level features using large scale unsupervised learning. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013. 8595–8598.

[32] Recht B, Re C, Wright S, et al. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. Advances in Neural Information Processing Systems, 2011. 693–701.

[33] Zhang S, Zhang C, You Z, et al. Asynchronous stochastic gradient descent for dnn training. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013. 6660–6663.

[34] Hinton G E, Salakhutdinov R R. Reducing the dimensionality of data with neural networks. Science, 2006, 313(5786):504–507.

[35] Grezl F, Karafiát M, Kontár S, et al. Probabilistic and bottle-neck features for lvcsr of meetings. Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on, volume 4. IEEE, 2007. IV–757.

[36] Yu D, Seltzer M L. Improved bottleneck features using pretrained deep neural networks. INTERSPEECH, volume 237, 2011. 240.

[37] Sivadas S, Hermansky H. Hierarchical tandem feature extraction. Acoustics, Speech, and Signal Processing (ICASSP), 2002 IEEE International Conference on, volume 1. IEEE, 2002. I–809.

[38] Dahl G E, Yu D, Deng L, et al. Context-dependent pre-trained deep neural networks for large-vocabulary speech recognition. Audio, Speech, and Language Processing, IEEE Transactions on, 2012, 20(1):30–42.

[39] Deng L, Li J, Huang J T, et al. Recent advances in deep learning for speech research at Microsoft. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on, 2013.

[40] Yu D, Seide F, Li G, et al. Exploiting sparseness in deep neural networks for large vocabulary speech recognition. Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on. IEEE, 2012. 4409–4412.

[41] Attwell D, Laughlin S B. An energy budget for signaling in the grey matter of the brain. Journal of Cerebral Blood Flow & Metabolism, 2001, 21(10):1133–1145.

[42] Lennie P. The cost of cortical computation. Current biology, 2003, 13(6):493–497.

[43] Moody J, Hanson S, Krogh A, et al. A simple weight decay can improve generalization. Advances in neural information processing systems, 1995, 4:950–957.

[44] Ranzato M, Boureau Y, LeCun Y. Sparse feature learning for deep belief networks. Advances in Neural Information Processing Systems (NIPS), 2007. 1185–1192.

[45] Dayan P, Abbott L F. Theoretical neuroscience. Cambridge, MA: MIT Press, 2001.

[46] Glorot X, Bordes A, Bengio Y. Deep sparse rectifier networks. Proceedings of the 14th International Conference on Artificial Intelligence and Statistics. JMLR W&CP Volume, volume 15, 2011. 315–323.

[47] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann machines. Proceedings of the 27th International Conference on Machine Learning (ICML-10), 2010. 807–814.

[48] Maas A L, Hannun A Y, Ng A Y. Rectifier nonlinearities improve neural network acoustic models. Proc. ICML, volume 30, 2013.

[49] Dugas C, Bengio Y, Bélisle F, et al. Incorporating second-order functional knowledge for better option pricing. Advances in Neural Information Processing Systems, 2001. 472–478.

[50] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 2014, 15(1):1929–1958.

[51] Sietsma J, Dow R J. Neural net pruning-why and how. Neural Networks, 1988., IEEE International Conference on. IEEE, 1988. 325–333.

[52] LeCun Y, Denker J S, Solla S A, et al. Optimal brain damage. Advances in Neural Information Processing Systems (NIPS), volume 2, 1989. 598–605.

[53] Hassibi B, Stork D G. Second order derivatives for network pruning: Optimal brain surgeon. Advances in Neural Information Processing Systems (NIPS), 1993. 164–164.

[54] Langford J, Li L, Zhang T. Sparse online learning via truncated gradient. The Journal of Machine Learning Research, 2009, 10:777–801.

[55] Povey D, Ghoshal A, Boulianne G, et al. The kaldi speech recognition toolkit. 2011..

[56] Graves A, Liwicki M, Fernández S, et al. A novel connectionist system for unconstrained handwriting recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2009, 31(5):855–868.

[57] Graves A, Mohamed A R, Hinton G. Speech recognition with deep recurrent neural networks. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013. 6645–6649.

[58] Sak H, Senior A, Beaufays F. Long short-term memory based recurrent neural network architectures for large vocabulary speech recognition. arXiv preprint arXiv:1402.1128, 2014..

[59] Graves A, Jaitly N. Towards end-to-end speech recognition with recurrent neural networks. Proceedings of the 31st International Conference on Machine Learning (ICML-14), 2014. 1764–1772.

[60] Hannun A, Case C, Casper J, et al. Deepspeech: Scaling up end-to-end speech recognition. arXiv preprint arXiv:1412.5567, 2014..

[61] Bengio Y, Simard P, Frasconi P. Learning long-term dependencies with gradient descent is difficult. Neural Networks, IEEE Transactions on, 1994, 5(2):157–166.

[62] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. arXiv preprint arXiv:1211.5063, 2012..

[63] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation, 1997, 9(8):1735–1780.

[64] Chan W, Lane I. Deep recurrent neural networks for acoustic modelling. arXiv preprint arXiv:1504.01482, 2015..

[65] Graves A, Jaitly N, Mohamed A R. Hybrid speech recognition with deep bidirectional lstm. Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on. IEEE, 2013. 273–278.

[66] Martens J, Sutskever I. Learning recurrent neural networks with hessian-free optimization. Proceedings of the 28th International Conference on Machine Learning (ICML-11), 2011. 1033–1040.

[67] Sutskever I, Martens J, Dahl G, et al. On the importance of initialization and momentum in deep learning. Proceedings of the 30th International Conference on Machine Learning (ICML-13), 2013. 1139–1147.

[68] Hinton G E, Vinyals O, Dean J. Distilling the knowledge in a neural network. NIPS 2014 Deep Learning Workshop, 2014.

[69] Bucilua C, Caruana R, Niculescu-Mizil A. Model compression. Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 2006. 535–541.

[70] Vanhoucke V, Senior A, Mao M Z. Improving the speed of neural networks on cpus. Proc. Deep Learning and Unsupervised Feature Learning NIPS Workshop, 2011.

[71] Vanhoucke V, Devin M, Heigold G. Multiframe deep neural networks for acoustic modeling. Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013. 7582–7585.

[72] Mohri M, Pereira F, Riley M. Weighted finite-state transducers in speech recognition. Computer Speech & Language, 2002, 16(1):69–88.

[73] Ljolje A, Pereira F, Riley M. Efficient general lattice generation and rescoring. Sixth European Conference on Speech Communication and Technology, 1999.

[74] Hori T, Hori C, Minami Y, et al. Efficient wfst-based one-pass decoding with on-the-fly hypothesis rescoring in extremely large vocabulary continuous speech recognition. Audio, Speech, and Language Processing, IEEE Transactions on, 2007, 15(4):1352–1365.

致　谢

感谢导师郑方老师在研究生期间对本人的悉心指导，他的言传身教将使我终身受益。

感谢王东老师对本人的耐心指导与帮助。王老师严谨的治学态度给我留下了深刻的印象，他的指导使我对学术研究有了更深刻的理解。

感谢实验室全体老师和同学们对我在学习、生活上的帮助与支持，特别是张之勇，殷实和刘荣同学，与你们在一起度过的这段时光我将终生难忘。

最后，感谢我的父母和我的朋友们，感谢你们长期以来对我无条件的关怀与支持。

声　明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，独立进行研究工作所取得的成果。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。

签　名：_____ 日　期：_____

个人简历、在学期间发表的学术论文与研究成果

个人简历

1989 年 10 月 27 日出生于北京市。

2008 年 9 月考入清华大学计算机科学与技术系计算机科学与技术专业，2012 年 7 月本科毕业并获得工学学士学位。

2012 年 9 月免试进入清华大学计算机科学与技术系攻读硕士学位至今。

发表的学术论文

[1] Liu C, Zhang Z, Wang D. Pruning deep neural networks by optimal brain damage. Fifteenth Annual Conference of the International Speech Communication Association, 2014.(EI 收录, 检索号:20144600199705.)

[2] Liu C, Wang D., Tejedor J. N-gram FST indexing for spoken term detection. Thirteenth Annual Conference of the International Speech Communication Association, 2012.(EI 收录，检索号:20132416407721.)

[3] Yin S, Liu C, Zhang Z, et al. Noisy Training for Deep Neural Networks in Speech Recognition. EURASIP Journal on Audio, Speech, and Music Processing, 2015.

[4] Meng X, Liu C, Zhang Z, et al. Noisy Training for Deep Neural Networks. IEEE China Summit and International Conference on Signal and Information Processing, 2014.