

分类号: TM391

单位代码: 10335

密 级: 绝密

学 号: 21221234

浙江大学

硕 士 学 位 论 文



论文题目 论文题目第一行
论文题目第二行

作者姓名 author

指导教师 _____

学科 (专业) 计算机科学与技术

所在学院 计算机科学与技术学院

提交日期 二〇一五年四月三十日

A Dissertation Submitted to Zhejiang
University for the Degree of
Master of Engineering



TITLE: English Title The 1st Line
English Title The 2nd Line

Author: Author

Supervisor:

Subject: Computer Science and Technology

College: College of Computer Science and Technology

Submitted Date: 2015-05-05

论文题目第一行

论文题目第二行



论文作者签名: _____

指导教师签名: _____

论文评阅人 1: _____ 评阅人 教授 浙江大学

评阅人 2: _____ 评阅人 教授 浙江大学

评阅人 3: _____ 评阅人 教授 浙江大学

评阅人 4: _____ 评阅人 教授 浙江大学

评阅人 5: _____ 评阅人 教授 浙江大学

答辩委员会主席: _____ 委 员 教授 浙江大学

委员 1: _____ 委 员 教授 浙江大学

委员 2: _____ 委 员 教授 浙江大学

委员 3: _____ 委 员 教授 浙江大学

委员 4: _____ 委 员 教授 浙江大学

委员 5: _____ 委 员 教授 浙江大学

答辩日期: _____ 2015-05-15

English Title The 1st Line

English Title The 2nd Line



Author's signature: _____

Supervisor's signature: _____

External Reviewers:

Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization

Examining Committee Chairperson:

Name	Professional Title	Organization
------	--------------------	--------------


Examining Committee Members:

Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization
Name	Professional Title	Organization

Date of oral defence: _____ 2015-05-15

浙江大学研究生学位论文独创性声明

本人声明所呈交的学位论文是本人在导师指导下进行的研究工作及取得的研究成果。除了文中特别加以标注和致谢的地方外，论文中不包含其他人已经发表或撰写过的研究成果，也不包含为获得 浙江大学 或其他教育机构的学位或证书而使用过的材料。与我一同工作的同志对本研究所做的任何贡献均已在论文中作了明确的说明并表示谢意。


学位论文作者签名： 


签字日期： 2015 年 3 月 10 日

学位论文版权使用授权书

本学位论文作者完全了解 浙江大学 有权保留并向国家有关部门或机构送交本论文的复印件和磁盘，允许论文被查阅和借阅。本人授权浙江大学可以将学位论文的全部或部分内容编入有关数据库进行检索和传播，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。

（保密的学位论文在解密后适用本授权书）

学位论文作者签名： 

导师签名： 

签字日期： 2015 年 3 月 10 日

签字日期： 2015 年 3 月 10 日

摘要

脑机接口 (Brain Machine Interface, BCI) 是一个完成大脑神经元和外部机器互联, 通信与控制的接口。脑机接口可以代替受损的神经系统, 通过大脑信号采集, 信号处理, 解码为计算机指令, 反馈这四步, 为残障人士提供自动化服务。本论文主要研究植入式脑电信号处理, 压缩, 以及 EEG 信号中的 P300 波形检测。

植入式脑电信号具有高采样率, 多通道, 高分辨率的性质, 因此压缩对于信号存储和传输都是必不可少的环节。在压缩方面, 过去的神经信号压缩工作主要关注于非植入式脑电信号, 这些方法都应用了相关信号的特性。但这些信号同植入式信号有很大差异, 它们的有效信息都在低频部分, 可以直接通过低通滤波器获得有效信号, 而植入式脑机接口获得的信号中高频部分包括神经元锋电位 (spike), 它能有效地进行解码, 因而不能丢弃。所以, EEG 等信号的压缩方法不能直接套用在植入式脑机接口获得的信号中。本文通过植入式电极研究大脑运动皮层信号性质, 利用信道内部神经信号特性建立了一个完整的高保真运动皮层信号压缩框架。

另一方面, 低分辨率的非植入式脑机接口采样方便, 在信号解码中广泛应用。只有正确地解码才能实现 BCI 执行符合用户意愿, 所以神经解码是 BCI 的核心任务, 我们在本文进行解码的前一步工作, P300 波形信号处理与检测。本文中, 我们采用深度学习的方法应用于 P300 波形, 分别用卷积神经网络和循环神经网络对 EEG 信号建模, 结合 EEG 信号在时间和空间维度的特性提高检测准确性, 完善模型。

关键词: 脑机接口, 压缩, 卷积神经网络, 长短期记忆方法, 循环神经网络

Abstract

Brain Machine Interface (BMI) is an interface that connect neurons of brain and devices to communicate and control. BMI can substitute impaired neural system, provide automatic service to disabled people through four steps: sampling signal from brain, signal processing, signal decoding to computer instruction, and feedback. This thesis focus on invasive electroneurographic signal processing, compression and P300 wavelet detection in Electroencephalograph signal.

Invasive electroneurographic signal with high sampling rate and multiple channel can represent high-resolution information. However, the huge quantity of data makes compression indispensable. In the aspect of signal compression, some related work focus on non-invasive electroneurographic signal. These methods take the consideration of signal characteristics. However, power centralizes in low frequency for non-invasive neural signal, therefore signal can be compressed through low pass filter directly. But the high frequency part of invasive BCI includes action potential (spike), which can be decoded effectively. As a consequence, traditional compression methods cannot be directly used in invasive BCI. In this thesis, we explore the characteristics of electroneurographic signal from motor cortex and provide a high fidelity compression framework for it.

On the other hand, non-invasive BMI provides low-resolution data easily, like Electroencephalograph, which is popular in neural signal decoding. As a pre task of machine execution, neural signal decoding is the core of BCI. Here we process and detect P300 wave, which is a pre-work of EEG signal decoding. In this thesis, we apply deep learning method to P300 wavelet detection, build convolutional neural network and recurrent neural network for EEG signal, and boost detection accuracy from both temporal and spatial dimension.

Keywords: Brain Machine Interface, compression, Convolutional neural network, Long Short Term Memory, Recurrent Neural Network

目录

摘要 i

Abstract..... ii

第 1 章 神经科学概述 1

 1.1 脑机接口介绍 1

 1.2 大脑的概念感知 2

 1.3 P300 信号检测 3

 1.4 文章结构 5

第 2 章 神经信号压缩 6

 2.1 神经电信号（electroneurographic signal） 6

 2.2 运动皮层胞外信号特点 7

 2.3 提出方法概括 9

 2.4 双阶编码 11

 2.4.1 符号编码..... 11

 2.4.2 量化..... 12

 2.5 混合无损编码 13

 2.5.1 Huffman 编码 13

 2.5.2 混合无损编码分界..... 15

 2.5.3 编码格式化..... 20

 2.6 实验 21

 2.6.1 数据集..... 21

 2.6.2 评价标准..... 21

 2.6.3 参数设置..... 23

 2.6.4 符号编码有效性..... 25

 2.6.5 方法比较..... 26

 2.6.6 计算代价..... 27

 2.7 结论 27

第 3 章 卷积神经网络分析信号	29
3.1 深度学习介绍	29
3.1.1 传统机器学习方法与局限性	29
3.1.2 深度学习方法介绍	30
3.1.3 深度神经网络结构与训练	33
3.2 卷积神经网络	39
3.2.1 CNN 网络结构	40
3.2.2 CNN 网络训练	44
3.3 ConvP300Net	45
3.3.1 数据集	45
3.3.2 相关工作	45
3.3.3 P300 检测网络结构	46
3.4 实验结果及实现	49
3.4.1 PConv 的学习	49
3.4.2 实验结果	50
3.4.3 实现方法	52
第 4 章 基于循环神经网络的信号解码	55
4.1 循环神经网络 (RNN)	55
4.2 LSTM 的应用	57
4.3 循环神经网络网络结构	57
4.3.1 RNN 网络结构	57
4.3.2 LSTM 的网络结构	58
4.4 LSTMP300Net	60
第 5 章 总结	61
参考文献	62
致谢	75

图目录

图 1.1	BCI 框架.....	2
图 1.2	可视化刺激任务的大脑通路	3
图 1.3	P300 刺激信号. 标准刺激 (S) 中的异常刺激 (T)	4
图 1.4	oddball task 实验矩阵.....	4
图 2.1	频域前 12 个系数的能量分布。	8
图 2.2	神经电信号高频部分 DCT 系数幅值分布。	9
图 2.3	通道间相关系数。	10
图 2.4	Flow diagram of the overall compression algorithm.....	10
图 2.5	频域量化系数。	14
图 2.6	混合编码格式示例	14
图 2.7	16 个序列段的零元素分布情况。	15
图 2.8	压缩数据格式化	20
图 2.9	截断数据段的两个重构效果对比示意图。	22
图 2.10	不同 T_{LH} 和 QT 下的 SNR, Spike Ratio 和 Compression Ratio。	24
图 2.11	不同量化表比例系数下的压缩效果。	24
图 2.12	不同块大小 S_b 下进行数据分割的压缩预处理的压缩结果。	25
图 2.13	我们方法与音频编码方法压缩效果比较。	27
图 3.1	Perceptront 图例.....	30
图 3.2	各层次 feature	31
图 3.3	稳定特征生成方法	32
图 3.4	前向网络的网络结构	34
图 3.5	误差反向传播示意图	36
图 3.6	$\frac{\partial L}{\partial net_j}$ 链式法则示意图.....	36
图 3.7	局部连接示意图	41
图 3.8	stride 示意图	42
图 3.9	LeNet5, 一个为手写数字做分类的卷积神经网络。 ^[1]	43
图 3.10	卷积神经网络结构示意图	44
图 3.11	P300 检测——数据段分割	47

图 3.12 P300 检测——数据预处理 47

图 3.13 ConvP300Net..... 48

图 3.14 激励函数 sigmoid,tanh,ReLu,softplus 50

图 3.15 ConvP300Net 训练 loss-迭代次数 53

图 3.16 ConvP300Net 测试 loss-迭代次数 54

图 3.17 ConvP300Net 测试准确率-迭代次数 54

图 4.1 LSTM 模型..... 59

图 4.2 LSTM 混合模型..... 60

表目录

表 2.1 SNR, Spike Ratio and Compression Ratio With and Without Symbol Encoding
(Low-Amplitude)..... 25

表 2.2 无损压缩方法性能比较 26

表 3.1 深度学习与传统模式识别方法 29

表 3.2 各类数据数目 45

表 3.3 实验人 A ——P300 检测各卷积神经网络结果 52

表 3.4 实验人 B ——P300 检测各卷积神经网络结果..... 52

第1章 神经科学概述

1.1 脑机接口介绍

由于现代计算机技术和神经科学学科的迅速发展，人们已经可以将大脑中的运动与计算机设备相关联，通过机器捕捉大脑中神经元的活动^[2]。这种为大脑和外部设备之间建立通路的方法与应用统称为脑机接口（brain computer interface，或 BCI）^[3;4]，以探索大脑活动与特定神经状态的关系。脑机接口可以通过测量脑信号从而理解主体意愿，而无需做出任何动作指示^[5-7]。因此 BCI 对残障人士或运动皮层受损，不能很好地与肌肉通信的病人有很大辅助作用。比如对于脊侧索硬化^[6]病人，如果可以有效地对脑信号解码，分析其意愿，就可以通过外设完成相应操作，从而辅助病人。

现在有很多测量脑信号的技术，如 fMRI（functional magnetic resonance imaging，功能性磁共振成像），NIRS（near-infrared spectroscopy，近红外光谱学），EEG（Electroencephalograph，脑电图），MEG（Magnetoencephalography，脑磁图）等。针对不同的采集信号，其信号预处理方法也各不相同。但是 BCI 最基本的任务都是正确地识别次级类别并翻译成机器指令以完成用户的意愿。脑机接口整个过程技术如图1.1所示，一个 BCI 需要包括：

1. 记录大脑活动

第一步，需要用放大器采集脑信号。

2. 提取并处理脑信号

第二步，需要将脑信号进行预处理，并分析其属于哪一类刺激，即进行信号解码。该步骤的解码部分非常重要，包括脑信号特征提取并翻译的过程。如果解码错误，后面的两步就会变得无意义了。

3. 将脑信号翻译成计算机指令

第三步，将解码后的信号发送到外部设备，执行相关指令。

4. 最后返回给用户

最后，外部设备将结果返回给用户。

图中，签名表示大脑神经信号的特定状态。

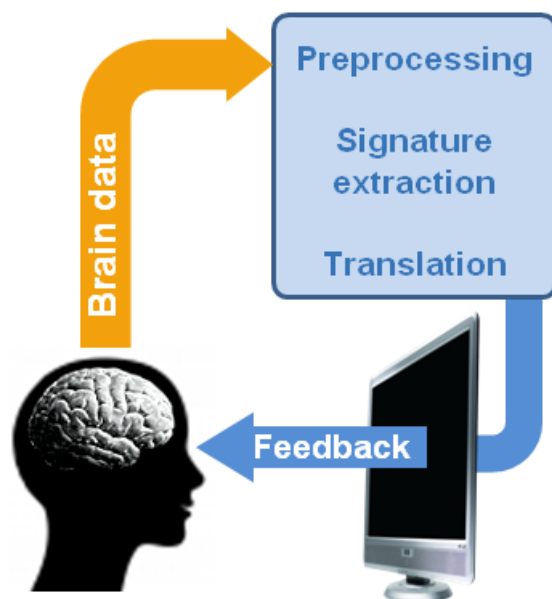


图 1.1 BCI 框架

在第二步的解码中，采用一个信号分类器，对不同刺激产生的信号进行分类。已有很多方法针对信号解码提出一些的模式识别方法。大多数有效方法基于机器学习模型^[8-10]，如支持向量机（SVM）^[1]，隐马尔科夫模型（HMM）和神经网络模型都曾被用于神经解码。此外，W.Wu 等人基于卡尔曼滤波器建立了一个实时解码系统，对运动皮层手臂区域大约 40 个神经元上所采集到的 spike 活动进行解码，并证明其效果好于之前用到的线性滤波器技术^[11]。在根据神经信号进行手臂路径跟踪方面，Gyron M. Yu 在模型复杂度和分类效果上做了一个平衡，提出了一个模型，将一些简单轨迹模型结合到一个估计轨迹的概率混合模型中，提高了分类准确率^[12]。对于类似的手运动轨迹方向估计，Caleb Kemere 等人基于学习的方法为运动轨迹建立了一个模型，并用最大似然估计进行参数求解^[13]。T. Horikawa 等人通过学习人在睡觉时候的 fMRI 与文字描述之间的联系进行神经信号解码^[14]。Lotte 对基于 EEG 信号的脑机接口分类技术做了总结^[9]。

1.2 大脑的概念感知

猴子对复杂的可视化刺激可以快速反应，平均反应时间在 250-260ms，最短可以达到 180ms。因此在试验中，我们建立了一套训练系统，采集猕猴运动皮层的神经电信号。如图 1.2 所示为一个基于视觉刺激，从视网膜到肌肉执行操作的有可能的大脑通路图^[15]。信息从视网膜传递到背外侧膝状体核（lateral geniculate nucleus, LGN），经过延迟传递到 V1（主要的视觉皮层）。紧接着，信息经过 V2 区，V4 区，再到后、前颞皮质区（Inferotemporal

cortex), 进行物体（高层）特征分析与描述。后颞皮质将信息映射到不同区域，包括可以进行物体分类的前额皮质（prefrontal cortex cortex, PFC）。为了将理解的命令传给肌肉执行，PFC 又把信息通过运动前皮质区（premotor cortex, PMC）和运动皮质 (motor cortex, MC) 传递给脊髓的运动神经元。最后，脊髓的运动神经元受刺激从而触发肌肉运动。图中，每个信息处理过程后有一个以毫秒为单位的时间，表示信息处理的估计时间，这样的反应速度也决定了我们采集信号的频率。

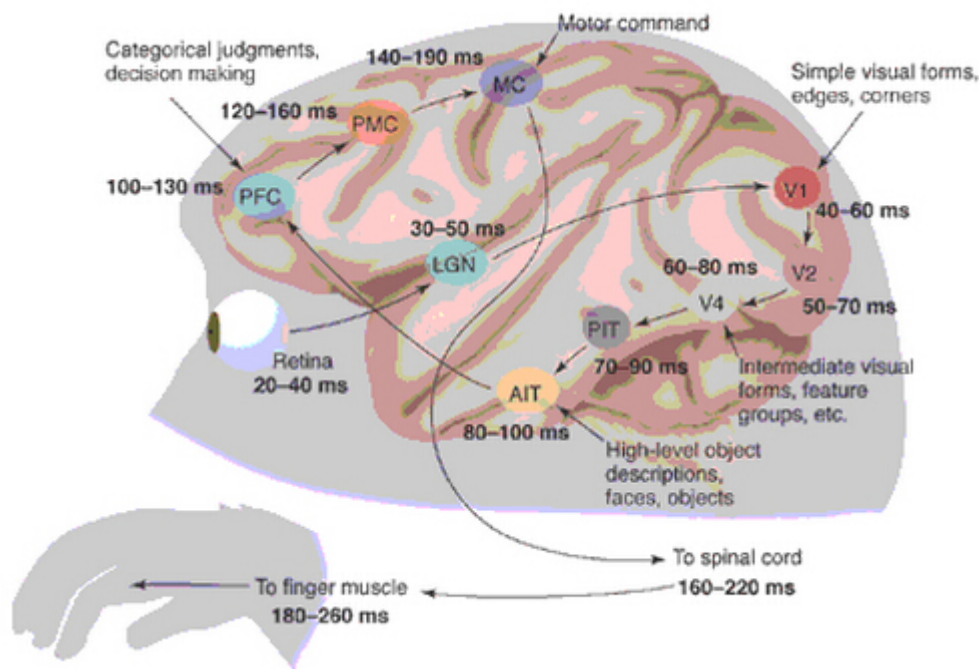


图 1.2 可视化刺激任务的大脑通路

1.3 P300 信号检测

P300 信号是人脑 EEG 信号中的波形，受任务相关事件刺激而产生。激发出 P300 信号的经典方法是通过“oddball task”产生，即在一个任务中出现两种模式随机出现，一种是频繁出现的，另一种是很罕见的模式。在任务中，给定一个刺激，并指导试验个体给出刺激所属模式，同时采集试验个体的 EEG 信号，罕见模式出现时对应的脑信号为 P300 信号。例如，在字符拼写任务中，oddball task 是一个关于字符注意的实验（如图1.3），展现一串字符（如 SSTSSSTSS），其中出现频率高的称为标准刺激（如该信号序列中的 S），频率低的称为异常刺激（如该信号序列中的 T）。当出现一个异常刺激时，300ms 后就会在 EEG 信号中产生一个正向偏移点位。这里标准刺激和异常刺激的差异可以用来识别所给刺激的类别，然后基于刺激发送信号给计算机指令执行。



图 1.3 P300 刺激信号. 标准刺激 (S) 中的异常刺激 (T)

1988 年 L. A. Farwell 和 E. Donchin 第一次用 oddball task 建立 BCI 系统^[16]。该系统中，将 36 个字符展现在一个 6*6 的矩阵中，矩阵的行和列随机闪烁，同一时刻只有一行或一列亮起，具体哪一行或哪一列是随机的。如图1.4所示，当一行一列相继闪烁的交点为指定字母时，测试者可以集中精神在头脑中进行简单计数或者确认，相应就会有 P300 在该位置产生，可以通过估计算法检测到这个 P300 信号。也就是说，如果在对应字母闪烁 300ms 后在某个位置检测到了 P300 信号，就说明主体正在注意这个特定位置。所以 P300 的检测实际上就是主体关注位置点字符的检测。



图 1.4 oddball task 实验矩阵

在 P300 检测中，需要先后两个过程。第一步，P300 位置检测。这是一个二类分类问题，给定一个时间段的信号，判断是否为 P300 波形。这个是在实验中进行标准的。比如，主体确定了当前要关注的字符，给定固定顺序随即闪烁行列的矩阵，我们就可以根据闪烁次序知道当前是否闪烁指定字符，即是否预期产生 P300 波形。但由于一些人为因素干扰，测试主体不一定能在指定时刻真的给出 P300 响应，所以第一步的目标是对这个带噪声的数据进行分类。第二步根据第一步的分类结果，对是 P300 的信号分类属于 36 个字符中的

哪一个。

1.4 文章结构

本文中，我们将分别用到两种 BCI 采集到的信号。脑机接口可以分为植入式和非植入式，非植入式方法，如头皮电信号（scalp electroencephalogram, EEG）易于获取，但是信号精度很差，采样率也相应很低。相反，植入式脑机接口用外科手术的方法将电极植入大脑皮层进行信号采集，可以采集到很高精度的细胞外神经元信号。在单个神经元中这种高精度信号包含神经锋电位（spike），或者叫做动作电位（action potential）。当神经元被激发的时候就会在神经元膜上产生离子电流，导致细胞去极化（depolarize）并激发出一个 spike 信号。

本文的组织结构如下：在第二章中，我们针对植入式神经信号占用空间大，传输困难的问题对其进行压缩，提出了一个新颖的压缩框架，并在猕猴上采集的神经电信号数据上证明了其有效性。之后，我们在非植入式 EEG 信号上采用深度学习的方法进行 P300 检测中的第一个环节，检测一段信号是否是 P300 信号。第三章和第四章分别采用卷积神经网络和循环神经网络对其进行建模，并分析模型的有效性。

第2章 神经信号压缩

在植入式脑机接口 (invasive Brain Machine Interface) 中, 我们将多电极阵列 (multi-electrode array, MEA) 植入大脑皮层从而获取高质量的电神经信号。这种信号的采样率为 30kHz, 给数据存储和传输带来了重大负荷, 所以我们需要对数据进行压缩来降低数据量。在这一节中, 我们结合大脑运动皮层神经电信号的特性, 提出了一种高保真压缩算法。实验中, 我们将该算法应用于哺乳动物的大脑运动皮层信号, 相对原信号得到了 18% 的压缩率, 而且没有对信号重建产生明显影响。该方法的信噪比 (signal to noise ratio, SNR) 达到 36dB, 而且 spike 信号也保存下来 92%, 大幅超过已有工作的效果。本章工作发表在 IJCNN 2014^[1]。

2.1 神经电信号 (electroneurographic signal)

本章中我们将主要关注运动皮层神经信号。作为大脑皮层的一个重要部分, 运动皮层负责计划, 控制并执行人体主动行为。在运动皮层功能的相关研究中, 电极阵列所采集的多通道信号通常在每个通道进行分频。通道中信号低频部分 (截止频率在 100Hz) 对应神经信号的局部场电位 (local field potential, LFP), 而中频到高频部分对应于动作电位 (spikes)。LFP 主要源于前突触行为, 反映了很多树突行为的平均电流。与之相反, Spike 主要反映兴奋神经元的行为。LFP 和 spike 信号对神经解码都很重要。对于运动皮层而言, spike 通常的持续时间小于 1 毫秒, 因此需要用高分辨率设备进行信号采集。这里我们用多电极阵列刺入细胞去采集数以百计的感兴趣神经元的信号。哺乳动物运动皮层神经元信号通常以 20-30kHz 的频率采集 128 个通道, 以保证可以完好保存 spike 细节。这样, 以 16-bit 的 A/D 分辨率计算, 如果采样率为 30kHz, 那么 128 个通道的信号就会以 7.68MB/s 的速度进行采集。换句话说, 一小时内的信号量就积累到 28.8GB, 这对信号的存储和传输都带来了巨大挑战。所以, 我们要对信号进行压缩。

尽管 BMI 系统已经建立得比较完善了, 脑皮层胞外信号的记录并没有深入研究过。在 Electromyography (EMG) 和 Electroencephalography (EEG) 信号的压缩上有过一些相关工作^[2], 为了有效压缩, 他们都结合了所处理信号的信号特性。但是植入式胞外信号与之相差甚远。

现有多通道压缩算法从两种思路进行实现。一种是应用通道内特性对每个通道的信号分别进行压缩, 另一种是用通道间相关性同时对所有通道的信号进行压缩。从第一个思路

出发, Weber 等人通过基于小波的编码器对老鼠躯体感觉皮质 (S1 区) 进行压缩, 然后这种方法代价是丢掉了 25% 的 spike, 对于后期的信号还原和分析并不理想。Chen 等人对老鼠的 S1 区域进行研究, 通过自适应信号量化在信噪比保持 25db 的时候达到的压缩率高于 25%, 那么信号压缩率和信号质量都得不到保证。为了改善他们的工作, Chen 从第二种思路出发, 利用通道间的信号相关性, 在 25db 信噪比的情况下, 将压缩率降至 5%。然而, 以上几种方法都丢失了太多信号细节, 白费了采集来的高分辨率信号。

本章中, 我们提出了一个运动皮层胞外信号的高保真压缩框架。首先, 我们讨论这种信号的 3 个特性: 1) 信号能量集中在低频; 2) 离散余弦变换系数中的高频部分可能对英语 spike 的激活模式 3) 通道间相关性不稳定。根据特性 (2), 我们提出了一个新颖的幅值滤波器, 将离散余弦变换系数按幅值, 而不是按频率分为两部分。低幅值成分由一个符号编码方法进行编码来降低全局失真; 高幅值成分, 包含主要信息和 spike, 由另一步骤编码。这个步骤叫混合编码, 包含哈弗曼编码和一个新颖的零长编码。我们的主要工作如下:

- 设计了一种新颖的幅值滤波器, 它将离散余弦变换系数根据幅值分为两部分, 这避免了 spike 信息的丢失。

提出了一个符号编码方法, 用来对低幅值成分进行编码, 而不是简单丢弃。这有效避免了全局信号失真。

- 发明了一种合并哈弗曼编码和新颖的零长编码的混合编码方法, 用来对高幅值成分和低幅值成分的索引进行编码。由此, spike 信息得到了精准的结构化保存。

最后, 我们用一系列方法测试我们提出的压缩框架, 得到了平均信噪比 36db, 压缩比 18% 的效果, 而且 spike 保真率保持在 92% 以上, 保证了重构效果。

2.2 运动皮层胞外信号特点

为了在保证信号质量的同时进行有效压缩, 我们在本节中多通道胞外信号的特点。我们的数据将在实验部分进行详细描述。这里, 我们从通道内特点到通道间相关性, 总结出三个特点:

1. 信号能量集中在低频:

为了研究所记录信号的频域特性, 我们采用离散余弦变换 (discrete cosine transformation, DCT) 将数据先转换到频域。作为傅里叶变换的一个变种, DCT 系数得到的是一系列实数, 处理起来比傅里叶变换方便。变换后的 DCT 系数用 $x_i = [x_i^1, x_i^2, \dots, x_i^N]$

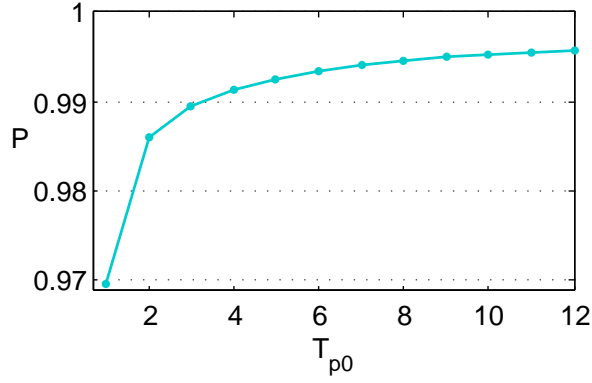


图 2.1 频域前 12 个系数的能量分布。

来表示，其中 x_i^j 表示第 i 个通道 DCT 系数中的第 j 个元素。那么整个数据集上低频部分能量比例为：

$$P = \frac{\sum_i \|x_i^1, x_i^2, \dots, x_i^{T_{p0}}\|_2}{\sum_i \|x_i\|_2} \quad \text{公式 (2.1)}$$

式中，分母表示所有通道总能量，分子为所有通道的前 T_{p0} 个 DCT 分量的总能量，即，以 T_{p0} 为截止频率，低频部分的能量。在整个数据集上，我们将 P 的平均值画在图2.1，图中横轴 T_{p0} 表示计算几个元素的能量和，纵轴表示前 T_{p0} 个元素的能量和占比，这清晰表明了少量 DCT 分量占据了信号的主要能量。换句话说，相当大的能量集中在了低频部分。

2. 高频信号中存在显著峰值：

和其它自然信号（如图像）一样，胞外信号的主要能力也集中在低频部分。然而，这样的信号在中高频有所差异。如图2.2所示为中频部分的一部分截断光谱，可见在 7325Hz 处有一个明显的峰值，对应于一个经常出现的神经元放电模式。实际上，实验表明很多通道共享这些具有峰值的频率，而一些通道没有。这可以从多电极阵列的采样原理理解，我们采集到的胞外信号的单通道信息可以由 3 至 5 个有不同 spike 激发模式的神经元组成。

3. 不稳定的通道间信号相关性：

在^[7]的工作中 Chen 等利用了老鼠 S1 区采集信号的通道间相关性达到了较高的压缩效果。受此激发，我们也来研究一下在哺乳动物的运动皮层是否存在这样的相关性。

将原信号分为 10 个等长连续段，用 $\mathbf{F} = \{\mathbf{F}^1, \mathbf{F}^2, \dots, \mathbf{F}^{10}\}$ 表示这 10 段在频域的平均 DCT 系数， $\mathbf{F}^i \in R^{N_c \times S_b}$ 表示第 i 段的平均 DCT 系数大小， N_c 为 channel 数目，

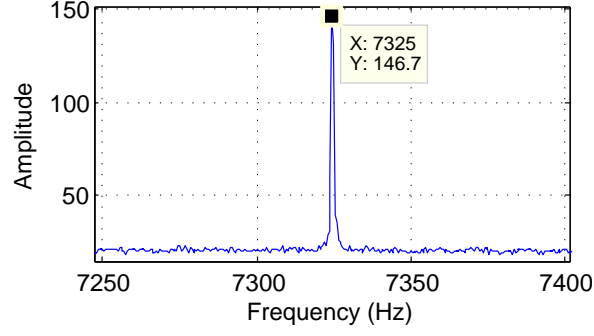


图 2.2 神经电信号高频部分 DCT 系数幅值分布。

S_b 表示待转换为频域的信号长度。对于每个 \mathbf{F}^i ，我们计算其两两通道间的相关系数，记 $\mathbf{C} \in R^{N_c \times N_c}$ 。图2.3(a)(b) 显示了一个时间序列段内频域的 96 个通道的两两相关系数矩阵，矩阵中，(i,j) 位置的值表示 i_{th} 通道和 j_{th} 通道之间的相关系数，越深表示相关性越大，可见相关性变化很大。用变换系数 (coefficient of variation) 衡量相关系数的相对离散程度：

$$CV = \frac{\sigma}{\mu} \quad \text{公式 (2.2)}$$

其中 σ 和 μ 分别表示相关系数的标准差和均值。当一个信号的 σ 相对 μ 可忽略不计时，即 CV 很小时，称信号为稳定信号。对每个时间段 \mathbf{F}^i , $\mathbf{CV} \in R^{N_c \times N_c}$ 衡量相关矩阵 \mathbf{C} 的浮动区间。CV 的均值显示在图2.3(c) 中，它是数据集中所有时间序列段内 CV 矩阵的平均值。(i,j) 位置的高度表示 CV 矩阵中 i_{th} 通道和 j_{th} 通道相关系数的平均值。这 $N_c \times N_c$ 个 CV 的均值为 0.68，也就是说，相关系数随时间剧烈变化，所以在我们获得的运动皮层神经信号中，通道间的相关性并不稳定，所以也较难将其应用在减少通道间冗余上。

2.3 提出方法概括

这篇文章中，我们考虑到上述信号特征，提出了一个高保真神经电信号压缩框架。首先，由于通道间相关性不稳定，我们对每个通道的信号做独立处理。整个框架的示意图见图2.4。它包括两个连续模块：“预处理”和“双阶编码”。

对于每个通道，首先将原信号分割成长度为 S_b 的块，然后每块通过以下两个模块进行处理：

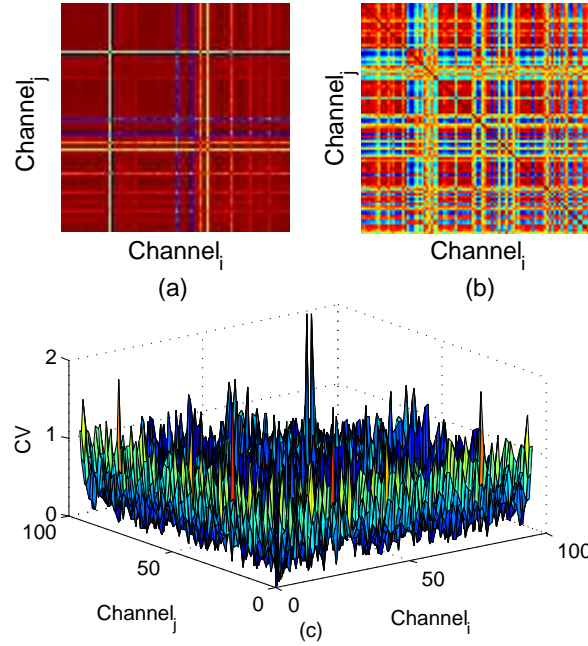


图 2.3 通道间相关系数。

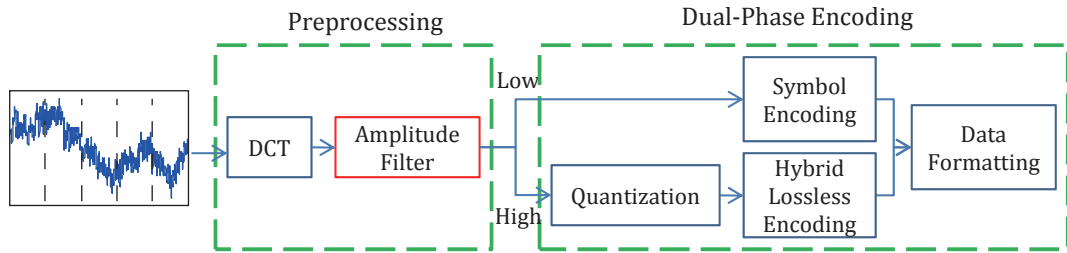


图 2.4 Flow diagram of the overall compression algorithm

a. *Preprocessing*

该模块通过离散余弦变换（Discrete Cosine Transformation, DCT）将原信号转换成频域。由于高频部分一些峰值可能对应于 spike firing 模式，所以传统压缩方法不适用于该信号的压缩。因此，我们提出按频谱幅值对信号进行分割，而不是根据频率高低分割。DCT 系数通过一个幅值滤波器，分为高幅值成分（High-Amplitude-Component, HAC）和低幅值成分（Low-Amplitude-Component, LAC）。HAC 包含显著地 LFP 和动作电位，其他信号归入 LAC。

b. 双阶编码

在神经信号的研究分析中已有证明，一个神经元完全由其神经元受刺激后的信号发放频率 firing rate 刻画^[17]。信号发放频率为单位时间内的动作电位发放个数。这个概念在 2007 年被^[18]中做出了修正，其中声明将 LFP 和 spike 结合起来可以加强神经信号解码准确率。也就是说，高幅值成分对神经信号表达更重要，因为这部分包含 LFP 和 spike

活动。相反，低幅值部分有效性较弱，但我们为了全局信号保真度和后续研究，也应在压缩时予以保存。这里我们提出双阶编码模块，它可以同时保存 HAC 和 LAC 的信息。在第一个阶段，低幅值部分由符号编码进行压缩，每个 LAC 中的 DCT 系数由 1bit 的符号表示；第二个阶段中，先将高幅值部分量化到一个相对小的范围区间，在通过我们提出的无损混合编码策略进行压缩。

2.4 双阶编码

双阶编码模块的提出就是为了在压缩过程中同时保存全局和局部神经电信号。它对预处理信号定义了两个独立的操作。第一个阶段采用有损压缩策略，通过符号编码压缩低幅值部分，在这个阶段，16bit 的原始信号值通过平均值平滑由 1bit 的符号表示。在第二个阶段，将 DCT 系数中 LAC 部分置零。所得到的向量中包含 HAC 部分量化后通过一个混合编码方法进行压缩。该编码方法将哈夫曼编码和我们提出的零长编码进行混合。哈夫曼编码处理高幅值项，而零长编码处理向量中的零元素。通过混合编码对 LAC 和 HAC 分开压缩可以有效地保存 LAC 和 HAC 的频谱位置，而无需存储多余信息。最后，对这两部分生成的编码进行格式化存储。

2.4.1 符号编码

在双阶编码模块的输入端，信号通过一个幅值滤波器。为了达到高压缩率，低幅值部分通常直接被舍弃。但这样的话大块丢弃的系数容易引发很大重构误差，从而影响信号保真度和后续研究。为了解决这个问题，我们为这部分提出了一个有效的表示方法。

在神经信号解码的研究中，从带噪声的数据中预测神经元响应是一大难题：给定相同的刺激，但神经元几乎没有两次会出现相同的活动模式。为了准确表示神经元表达，需要对信号做一些均值平滑处理^[19]。因此，均值通常用来代表原始系数，受这个激发，我们这里也在压缩过程中用信号的均值代替。

在符号编码中，均值在频域进行操作。低幅值 DCT 系数用其均值表示。由于正负系数进行平均可能被抵消，因此我们存储的是幅值，即系数的绝对值。为了有效压缩，各通道 LAC 部分的平均幅值预先计算好，然后用 1-bit 的符号表示其正负。记 LAC 中元素 i 为 l_i ，其符号表示为：

$$symbol(l_i) = \begin{cases} -1, & -T_{LH} < l_i \leq 0 \\ 1, & 0 < l_i < T_{LH} \end{cases} \quad \text{公式 (2.3)}$$

其中-1, 1 位系数符号, -1 示负, 1 表示正, T_{LH} 表示低幅值和高幅值之间的分界阈值。根据公式2.3, 含有 S_l 个元素的 LAC 向量通过符号编码被压缩为 S_l 个 bit 的向量, 用来在解码时恢复。

在压缩解码时, 我们将每个元素的编码符号乘以其幅值即可。这里有一个问题就是如何恢复 LAC 部分元素的位置。我们可以直接存储其位置, 但是这样太耗费空间。事实上, 我们无须显式存储其位置。由于 LAC 部分已经被编码, 我们可以在待压缩信号中将其置零, 而后续处理中只要保证其他压缩元素都非零即可, 这样, 在解码时只要找出所有零项即可恢复 LAC。

2.4.2 量化

由于 HAC 部分包含 LFP 和主要 spike, 这个重要部分要比 LAC 做更精细的保存。为此, 这一块信息线被量化到一个小区域, 然后通过一个混合编码方法进行压缩。本节中介绍第一步操作——量化。当信号从预处理模块输出是, 每个长为 S_b 的数据块通过一张量化表进行量化。这本身是一个有损压缩过程, 但是通过共享量化表可以丢掉部分冗余信息。量化定义为原始信号除以对应的量化值, 然后取整。这些量化值在不同信号通道又不同取值, 从而组成量化表 (Quantization Table, QT)。令 $QT \in R^{N_c \times S_b}$ 表示量化表, 其中 N_c 表示通道数, S_b 表示数据块大小。记量化后第 c 项为 H_c^Q , 有

$$H_c^Q = \text{round}(H_c ./ QT(c, :)), \quad \text{公式 (2.4)}$$

其中 $./$ 是一个向量逐项除法, $\text{round}(x)$ 为取整操作, 返回距离 x 最近的整数。注意, H_c 中的元素幅值都大于等于 T_{LH} , 所以只要能令 $QT(c, :)$ 中的元素都小于 T_{LH} , 那么量化后, HAC 的绝对值就都大于等于 1 了, 这样可以很好地分离压缩后信号中的 LAC 分量和 HAC 分量。

对于待量化的高幅值部分, 其取值范围由信号和 T_{LH} 共同决定。然而, 神经电信号会因个体差异而拥有不同的采集信号取值^[20]。而且, 不同神经元拥有不同的频谱分布, 因此需要对每个个体的不同通道建立特定量化表。在符号编码中我们已经计算了 LAC 的平均幅值, 这里我们将其用于量化表。

该量化表的设计出于以下几点考虑。一, 考略到个体差异性, 我们给每个实验个体建立独一无二的量化表。二, 我们在研究运动皮层神经电信号特性时讲到过不同通道具有不同皮普分布, 因此量化分量在通道间不共享。三, 由于取整操作, 转换后的取值有所变化,

但是变化幅度不会大于对应位置量化值的 $\frac{1}{2}$ ，不会损失太多信息。而且，这样量化表的值也可以都小于 T_{LH} ，使得量化后结果不小于 1，满足了??节中提到的非零特性。

2.5 混合无损编码

由于高幅值部分所有元素在量化后都成为了整数，离散分布的信号可以通过编码无损压缩，该无损压缩可由哈夫曼编码和零长编码方法实现。

作为最优符号编码方法，哈夫曼编码生成最佳可变长码，可以有效运用在我们量化后的数据上。然而，很多高频部分 DCT 系数的幅值都很小，容易被归入 LAC 分量。经过幅值过滤器处理后，很多高频参数在 HAC 中变成了 0。因此，可以通过将高频部分连续的 0 替换成 0 的个数达到更好的编码效果，这就是零长编码的思想。于是，所有非零 DCT 系数都可以与哈夫曼编码表示；设置一个界限 B ， B 之前的零用哈夫曼编码方法， B 之后的零用零长编码分别进行表示。为了有效结合这两种方法， B 的设定就很重要了，而它的取值取决于零的分布。为了更清晰地描述我们提出的混合编码方法，我们首先介绍一下哈夫曼编码和零长编码，然后来看怎样设置临界点 B 。

2.5.1 Huffman 编码

熵编码方法是一种无损压缩技术，通常为每个符号创建一个独一无二的码。作为最常用的熵编码方法，哈夫曼编码^[21]用在我们的无损编码部分，旨在建立一棵最优树，能够最小化加权树高和（即信号编码总长度）^[22]。为了将原始值转换成二进制序列，哈夫曼编码方法基于每个符号的出现次数进行编码。

首先来看计算 DCT 系数的分布。根据我们的实验，HAC 系数服从高斯分布，而很高的幅值非常少，那么为所有系数建立哈夫曼编码就是一件既耗时又浪费空间的做法，因为首先要计算其哈夫曼编码，而后由于其频率太少，会导致码字很长，浪费空间。因此我们并非计算所有 DCT 系数分布，对于非零系数，我们只计算 $[-Z, Z]$ 的部分，其中 Z 是系数的统计范围。而超出该范围的幅值，我们单独记录其值和位置。此外，并非所有系数零都予以统计。令 B 表示哈夫曼编码与零长编码之间的分界点，由于只有 B 之前的零采用哈夫曼编码，我们就只统计 B 之前出现的零的个数。

统计幅值在 $[-Z, Z]$ 内所有非零系数以及 BHz 前零元素的分布，将 N_c 个通道求得平均，如图2.5所示，其中横轴表示 $[-31, 31]$ 的整数系数范围，纵轴表示量化后每个系数的出现次数。在这个例子中， B 由??节中的定界策略决定。根据这个统计结果，哈夫曼编码对这些系数进行编码。

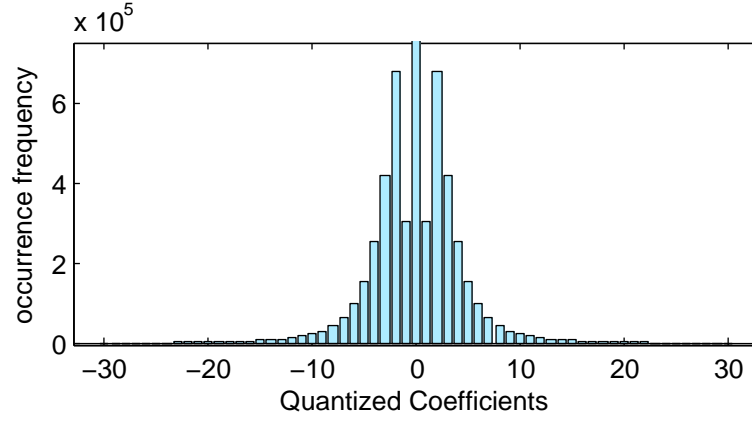


图 2.5 频域量化系数。

2.5.1.1 零长编码 (Zero-Length-Encoding)

零长编码方法为了对高频部分连续的零进行编码。我们用八进制表示法表示连续 k_z 个零，用哈夫曼编码来分隔两个相邻的八进制数字。举例， $k_z = (8A + B) \times 8 + C$ ，其中 A, B, C 分别表示三阶，二阶，一阶的八进制数字。用 **HCT** 表示哈夫曼码表 (Huffman Code Table)，即 DCT 系数的不定长编码表，混合编码格式如下：

图 2.6 混合编码格式示例

$$\underbrace{\quad\quad\quad}_A \text{HCT}(0) \underbrace{\quad\quad\quad}_B \text{HCT}(0) \underbrace{\quad\quad\quad}_C$$

令 $g(k)$ 为要表示的 0 个数所需阶数，我们有

$$g(k_z) = \begin{cases} 1, & k_z \in [0, 7] \\ 2, & k_z \in [8, 7 \times 8 + 7] \\ 3, & k_z \in [64, 63 \times 8 + 7] \\ \dots \end{cases}$$

可以规范化为：

$$g(k_z) = \begin{cases} 1, & k_z = 0 \\ \lceil \log_8(k_z + 1) \rceil, & \text{else} \end{cases} \quad \text{公式 (2.5)}$$

在解码过程中，首先读入一个 3-bit 二进制码字，然后检测下一个序列是否和 $\text{HCT}(0)$ 相同。如果是，就再读三位，以此类推，直到条件不满足为止，这时就可以计算该段有多少个连续 0 了。由于哈夫曼编码是前缀无歧义编码 (prefix-free code)，所以可以保证零的表示不会出现歧义。

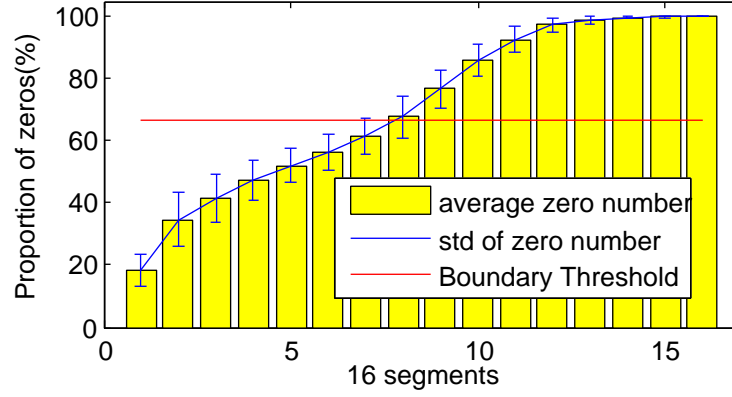


图 2.7 16 个序列段的零元素分布情况。

零长编码通过直接编码连续零，而非一个个表示来节省空间，因此我们希望零长编码的信号段拥有更多连续的零（而不是分散的），因此确定一个好的分界点 B 尤为重要。

2.5.2 混合无损编码分界

随着量化信号中连续零数量的增多，可以想见零长编码要比哈夫曼编码短，为了证实这个想法，我们来看混合编码策略中两种编码方法的编码。

对分界面的简单估计可以通过计算零的个数分布而得。将每 S_b 个离散时间序列点分为 N_s 个等长短序列段，然后计算每段中零的个数。图2.7显示了 1600 个 DCT 系数分割为 16 段 ($S_b=1600, N_s=16$) 的零数分布，对于 $S_b = 1600$, 将 1600 个元素分为 16 段，每段有 100 个分量。黄色和蓝色线条分别表示每段中零元素个数的均值和方差。可以看出零的个数分布随着频率而增多。用 $\Omega_0 \in R^{N_s}$ 表示平均零数分布，即图中的黄色条，则可以根据零数分布情况进行分界。

令 T_{HZ} 表示在哈夫曼编码与零长编码的分界 B 处，零的出现概率，那么该段中零的个数可以用 $T_{HZ} \cdot \frac{S_b}{N_s}$ 表示，其中 $\frac{S_b}{N_s}$ 是每段中的元素个数。这样，分界限可由式2.6得出， B 之前采用哈夫曼编码， B 之后采用零长编码。

$$B = \text{CrossSegment} \left(\Omega_0, T_{HZ} \cdot \frac{S_b}{N_s} \right) \cdot \frac{S_b}{N_s} \quad \text{公式 (2.6)}$$

该式中，CrossSegment 函数返回零的出现频率和阈值 T_{HZ} 的交点出数据段编号。图2.7中红色线表示阈值 $T_{HZ} * (S_b/N_s)$, 用来分隔混合编码中的两个算法, 图中可见零分布线（黄色）与阈值线（红色）交于第 7 段。该例中， $S_b=1600, N_s=16$, 定义 $T_{HZ}=0.75$, 那么有 $B=700$ 。

下面我们来仔细分析交点位置。用 x 表示交点段的编号，在式2.6中直接用 T_{HZ} 乘以每段中的元素个数，而在式2.7中，考虑到了交点段中阈值与段交点处所占的比例。

$$\begin{aligned} x &= \text{CrossSegment} \left(\Omega_0, T_{HZ} \cdot \frac{S_b}{N_s} \right) \\ B &= \left((x-1) + \frac{T_{HZ} \cdot \frac{S_b}{N_s} - \Omega_0(x-1)}{\Omega_0(x) - \Omega_0(x-1)} \right) \cdot \frac{S_b}{N_s} \end{aligned} \quad \text{公式 (2.7)}$$

最后一个问题是 T_{HZ} 的设定。由于 T_{HZ} 表示在 B 处，系数零所占比例，它可以表示每个非零元素之前平均零的个数，即

$$T_{HZ} = \frac{k_z(B)}{k_z(B) + 1} \quad \text{公式 (2.8)}$$

其中的 1 表示非零系数， $k_z(B)$ 表示在分界线 B 处一个非零系数之前的平均零的个数。

基于 DCT 系数的统计特性，分界可以进一步提升压缩性能。然而，从上面的推导可知，在式 Eq.2.7 和 2.8 中，计算 B 和 T_{HZ} 是一个死锁。公式 2.7 需要 T_{HZ} 的值，而 T_{HZ} 的值要在式 2.8 中由 B 决定。为此我们提出了一个迭代检验方法，该法需要根据混合编码两种方法生成的码字长度合理确定界限的位置。

令 HCT 表示前面生成的哈夫曼编码表， $HCT(x)$ 表示 x 的哈夫曼编码， $l_0 \in Z_+$ 表示 HCT 码表中 0 的码字长度， k_z 表示一个非零元素之前 0 的平均个数。若量化系数全部由哈夫曼编码表示，其长度为：

$$l_1 = \sum_{i=1}^I [HCT(x_i)] + l_0 \cdot k_z I, \quad x_i \in H_c^Q, x_i \neq 0 \quad \text{公式 (2.9)}$$

其中 I 是 H_c^Q 中所有非零系数集合的大小。该式中，第一项将所有非零元素 x_i 的长度进行加和，而 $l_0 \cdot k_z I$ 表示所有 ‘0’ 的长度和，因为每个非零系数之前平均有 $l_0 \cdot k_z$ 位数字。

类似地，如果量化系数中的零全部由零长编码表示的话，编码长度为：

$$l_2 = \sum_{i=1}^I [HCT(x_i) + (3 + l_0)g(k_z) - l_0], \quad x_i \in H_c^Q, x_i \neq 0 \quad \text{公式 (2.10)}$$

根据零长编码格式 2.5.1.1， $(3 + l_0)$ 是八进制表示中每多一阶多需要的表示位数。用 $g(k_z)$ 表示 k_z 个零所需阶数，则 $(3 + l_0)g(k_z) - l_0$ 可以表示一个非零元之前的零分量用零长编码所需长度。此处 k_z 并非确切平均零的个数，而是为了计算方便估计的最近整数。

To compare the two encoding length, we use Eq.(2.9) minus Eq.(2.10) and take the part in the bracket as $f(k_z)$,

为了比较两种编码方法的长度，我们用公式2.9减去式2.10，计入下式方括号内，即 $f(k_z)$ ：

$$l_1 - l_2 = [l_0 \cdot k_z - (3 + l_0)g(k_z) + l_0] I = f(k_z) \cdot I \quad \text{公式 (2.11)}$$

由于 I 是常数，我们只考虑以 k_z 为参数的函数 f 。将式 (2.5) 代入式 (2.11)，我们有

$$f(k_z) = \begin{cases} -3, & k_z = 0 \\ l_0 \cdot k_z - (3 + l_0) \lceil \log_8(k_z + 1) \rceil + l_0, & \text{else} \end{cases} \quad \text{公式 (2.12)}$$

对于离散整数 k_z ，相邻项之间的差为：

$$f(k_z) - f(k_z - 1) = \begin{cases} f(0) = -3; \\ f(k_z) = f(0) + l_0 k_z - (3 + l_0) \lceil \log_8 k_z \rceil \end{cases} \quad \text{公式 (2.13)}$$

$$\text{where } k_z \in Z_+, l_0 \in Z_+$$

由式 (2.13) 可得， $f(k_z)$ 只有在 k_z 最开始，即 l_0 不大于 1 时才会小于零。由于随着频率增加， $f(k_z)$ 有增加的趋势，所以 $f(k_z)$ 与 0 只有一个交点，就是在 $\lceil \log_8 k_z \rceil$ 等于零的时候。在这个交点处，我们可得从式 (2.13) 得：

$$k_z = \frac{3}{l_0} \quad \text{公式 (2.14)}$$

所以，混合编码中的阈值可以由式 (2.8) 和 (2.14) 确定。但是在参数选择过程中还有其他问题。正如我们之前提到的，在确定界限 B ，阈值 T_{HZ} 和哈夫曼编码表之中有死锁。问题在于， T_{HZ} 由 k_z (式 (2.8)) 计算而得，而 k_z 依赖 l_0 (式 (2.14))， l_0 又由哈夫曼码表确定。然而，**HCT** 需要计算 B 之前零的个数，而这又由 T_{HZ} 得来。为了打破这个死锁，我们再假设 $l_0 = 1$ 下初始化 k_z ，见 Boundary Descent 算法。

算法1 描述了界限下降算法的流程。令 ε 表示是否迭代的标志， $l_0(\text{HCT})$ 表示哈夫曼码表中零编码长度，从假设 $l_0(\text{HCT})$ 为 1 开始，算法初始化 k_z 为 3 (第一行)。只要迭代标志为真，就计算所有样本中的零分布 (3-7 行)。然后通过式 (2.7) 计算该方法的阈值 T_{HZ} (第 8 行)，然后根据式 (2.8) 计算混合编码中两种方法的分界点 B (9-10 行)。以 B 和量化信号 H_c^Q 作为输入，可获哈夫曼码表 **HCT** (line 11)。然后，检验最初的假设，即是否满足式 (2.14)。如果迭代所得 l_0 与假设不吻合，就用 $\frac{3}{l_0}$ 代替，始终迭代直到达到条件 $l_0(\text{HCT}) \cdot k_z = 3$ (12-16

Algorithm 1 BOUNDARY DESCENT ALGORITHM

Input: Quantized HAC to be compressed H_c^Q
Output: HCT, B

```

1:  $k_z \leftarrow 3, \varepsilon \leftarrow 1$ 
2: while ( $\varepsilon = 1$ ) do
3:   for  $c \leftarrow 1$  to  $N_c$  do
4:     for  $i \leftarrow 1$  to  $S_b$  do
5:        $\Omega_0(i) \leftarrow \sum_{c=1}^C 1_{\{H_c^Q[i]=0\}}$ 
6:     end for
7:   end for
8:    $T_{HZ} \leftarrow \frac{k_z}{k_z+1}$ 
9:    $x = \text{CrossSegment} \left( \Omega_0, T_{HZ} \cdot \frac{S_b}{N_s} \right)$ 
10:   $B = \left( (x-1) + \frac{T_{HZ} \cdot \frac{S_b}{N_s} - \Omega_0(x-1)}{\Omega_0(x) - \Omega_0(x-1)} \right) \cdot \frac{S_b}{N_s}$ 
11:   $HCT \leftarrow \text{HuffmanEncoding}(H_c^Q, B)$ 
12:  if ( $l_0(HCT) = \frac{3}{k_z}$ ) then
13:     $\varepsilon \leftarrow 0$ 
14:  else
15:    if ( $l_0(HCT) \leq 3$ ) then
16:       $k_z \leftarrow \frac{3}{l_0(HCT)}$ 
17:    else
18:       $B \leftarrow 0, \varepsilon \leftarrow 0$ 
19:    end if
20:  end if
21: end while

```

行)。注意 l_0 应被限制在 3 之内, 否则就无须用哈夫曼编码了 (17-19 行)。这个算法最终返回 **HCT** 和分界处 B 。

Algorithm 2 Overall Compression Algorithm

Input: \mathbf{X} , the signal; S_b , the block size; T_{LH} , the threshold between HAC and LAC; B , the boundary within Hybrid Encoding

Output: \mathbf{Y} , formatted compression result; \mathbf{Z} , lengths of *Symbol Encoding* codes for all blocks

```

1: Divide  $\mathbf{X}$  into blocks of size  $S_b$ ,  $\mathbf{X}_{(1)}, \mathbf{X}_{(2)}, \dots, \mathbf{X}_{(N)}$ 
2: for  $i = 1, \dots, N$  do
3:    $\mathbf{F}_{(i)} \leftarrow DCT(\mathbf{X}_{(i)})$ 
4:    $\mathbf{low}_{(i)} \leftarrow \text{find indices } (\mathbf{F}_{(i)} < T_{LH})$ 
5:    $\mathbf{LAC}_{(i)} \leftarrow F(\mathbf{low}_{(i)}); \%LAC$ 
6: end for
7:  $\mathbf{QT} \leftarrow \text{average over } |\mathbf{LAC}_{(i)}|, i = 1, \dots, N$ 
8:  $\mathbf{Y} \leftarrow []$ 
9: for  $i = 1, \dots, N$  do
10:   $\mathbf{HAC}_{(i)} \leftarrow \mathbf{F}_{(i)}; \mathbf{HAC}_{(i)}(\mathbf{low}_{(i)}) \leftarrow 0$ 
11:   $\mathbf{S} \leftarrow \text{sgn}(\mathbf{LAC}_{(i)}); \mathbf{Y} \leftarrow [\mathbf{Y} \ \mathbf{S}]; \%Symbol\ Encoding$ 
12:   $\mathbf{Z}_{(i)} \leftarrow \text{length}(\mathbf{S});$ 
13:   $\mathbf{H}_c^Q \leftarrow \text{round}(\mathbf{H}_c(i) ./ \mathbf{QT});$ 
14:   $\mathbf{H} \leftarrow \text{Huffman}(\mathbf{H}_c^Q(1 : B)); \mathbf{Y} \leftarrow [\mathbf{Y} \ \mathbf{H}];$ 
15:  for all  $x \in \mathbf{H}_c^Q((B + 1) : \text{end})$  do
16:    if  $x \neq 0$  then
17:       $\mathbf{Y} \leftarrow [\mathbf{Y} \ \text{Huffman}(x)]$ 
18:    else
19:       $\mathbf{Y} \leftarrow [\mathbf{Y} \ \text{ZeroLength}(x)]$ 
20:    end if
21:  end for
22: end for

```

通过算法1 求得压缩框架的全部参数后, 我们用算法2总结一下整个算法的流程。首先将信号以 S_b 为大小分割成 N 块, 对于每个数据块, 分别映射到频域, 求 DCT 系数, 求取 LAC 分量并按符号编码方法压缩 (2-6 行)。根据 LAC 分量幅值确定每个通道的量化表 QT

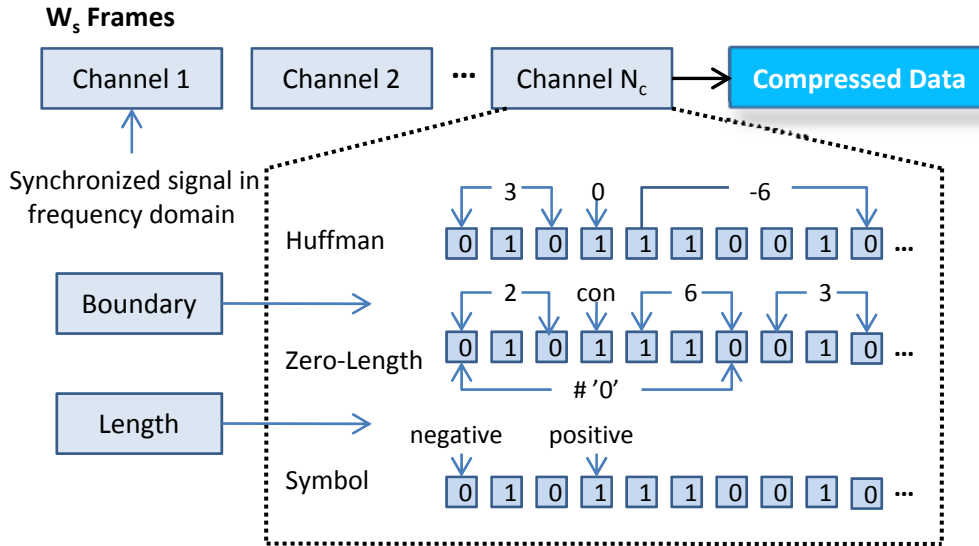


图 2.8 压缩数据格式化

(第 7 行)。然后对于每个块，对 LAC 分量编码，生成 HAC 分量并量化压缩，对混合编码分界点 B 之前的分量进行哈夫曼编码，对 B 之后的零分量进行零长编码，非零分量进行哈夫曼编码 (10-21 行)。最后返回编码 Y 。

2.5.3 编码格式化

本节中，我们讨论数据的格式化存储。在前面的与处理步骤中已经提到，每个通道的信号首先在时域上分成很多数据块，每个数据块包括 N_c 个通道，如图 2.8 所示。

根据三个二进制编码方法，每个通道的码字包括哈夫曼编码，零长编码和符号编码三部分结果。对于高幅值分量，假设哈夫曼码表中有 $HCT(0) = 1, HCT(3) = 010, HCT(-6) = 110010$ ，对哈夫曼编码部分，借助 HCT 进行解码，对零长编码部分，由于 k_z 个 0 用 $(3 + l_0)g(k_z) - l_0$ 位来记录，所以也可以根据表示位数进行解码。举个例子，加入我们现在拿到的编码结果为：“0101110”，那么在哈夫曼编码中，该码字表示的就是 3, 0, -6；而零长编码中，该码字则表示 22 ($22 = 2 \times 8 + 6$) 个零，后面跟一个 3。当我们将 HAC 部分解码完毕后， LAC 在最后进行解码，表示每个值的正负，如图 2.8 所示。

在编码过程中，由这三种编码而得的是一个二进制流。但是我们需要将这三种编码的结果分离开才能进行解码，因此在码字最开始的地方我们记录两个位置，一个是哈夫曼编码和零长编码的边缘，一个是零长编码之后，符号编码之前的分离点。此外，我们还需要记录的是每个个体的对应的量化表。

2.6 实验

这一节展现我们的实验结果，数据基于过去四年从两只猴子采集的数据。我们讨论了参数设定问题，并与经典信号压缩算法做了比较。

2.6.1 数据集

我们的实验数据及采集自浙江大学求是学院 BMI 系统^[23]。为了训练猴子，建立了一套训练系统，该系统中，每只猴子在一个从中心出发的四方向摇杆上做训练，目标是根据可视的提示将方向杆摇至正确方向。完成了这项任务后，会对猴子大脑运动皮层 (M1 区) 植入一个多电极阵列来捕捉手摇杆动作所产生的神经信号。每次试验持续大概 60 分钟。

这项任务在一个多通道捕获设备 (Cerebus 128TM (Blackrock Microsystem, Salt Lake City, UT, USA)) 上完成，同时记录 106 个神经元的信号。信号在 96 个电极 (电极长 1.0mm, 总长度 7.0cm) 上进行采样，即以 30kHz 为采样频率，采 96 个通道，16-bit 的分辨率。这样实验所获数据流为 5.76MB/s, 也就每 5 分钟获得 1.73GB 的数据。为了检验我们的压缩算法，我们随机选择了 12 条记录，每条记录长 300 秒。

2.6.2 评价标准

为保证神经电信号压缩后的可用性，压缩算法在减少信息所占空间的同时希望保证重构信号的相对信息完整^[24]。从这个角度出发，我们需要一些压缩的评价标准来度量压缩效果。

1. Signal to Noise Ratio

在信息论中，信噪比 (Signal to Noise Ratio, SNR) 用来评判信号压缩的保真度。

令 S_o 和 S_r 分别表示原始信号和重建新号, SNR 定义为 S_o 与 $S_o - S_r$ 的能量比:

$$SNR(S_o, S_r) = 10 \cdot \log_{10} \frac{\|S_o\|_2^2}{\|S_o - S_r\|_2^2} \quad \text{公式 (2.15)}$$

作为一个基于能量的评判标准，SNR 可以很好地反应误差的能量，但这还是不够的。回想我们在第??节中分析的第一条信号特性，信号的主要能量集中在低频区域。也就是说低频区域对 SNR 影响很大，而这对高频区域就不平等了。为了解决这个不平等问题，我们将在另外的评判标准中聚焦于 spike 信号，也就是高频部分的主要有效信号。

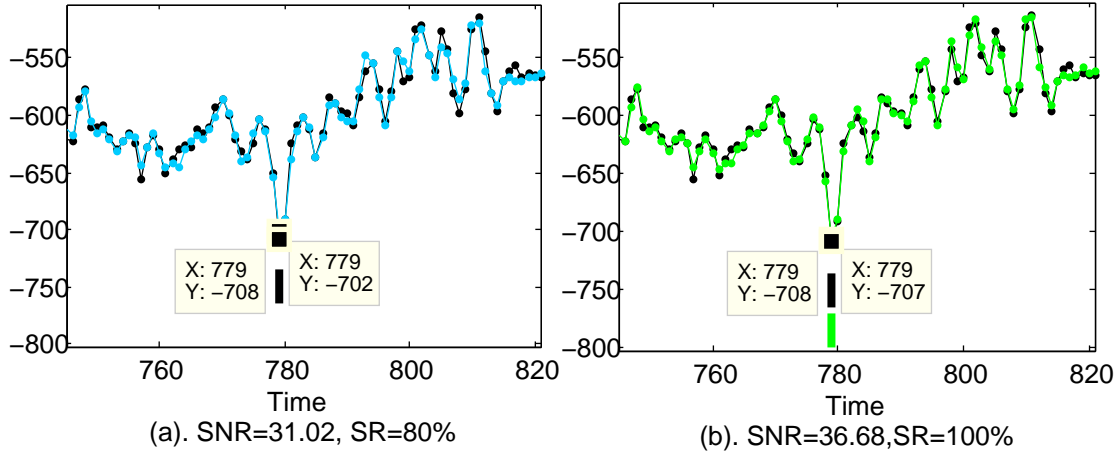


图 2.9 截断数据段的两个重构效果对比示意图。

2. Spike ratio

为了检验 spike 的保留程度，我们引入 Spike Ratio, 表示一段信号与其压缩后重建新号相比保留的 spike 比例。在我们的验证中，常用的幅值阈值技术^[25]用来检测 spike, 其中阈值 Thr 设定为:

$$Thr = \alpha \cdot \sigma_n, \sigma_n = median\left(\frac{|x|}{0.6745}\right) \quad \text{公式 (2.16)}$$

其中 α 是一个常量银子, σ_n 是背景噪声的标准差估计量. 如果一个点的值大于 Thr , 就视其为一个 spike 的起始点。注意计算 spike 不留程度不是一个计数问题，而是计算匹配的 spike 个数，也就是有多少 spike 保留在了正确的位置。在此过程中，我们分别在原始信号和压缩重建新号上检测 spike，并计算 Spike Ratio。

为了帮助理解重建效果，图??给出了一小段截取的数据在不同 SNR 和 Spike Ratio 上的压缩结果，其中黑线表示原始信号，蓝色和绿色线表示重构信号，检测到的显著 spike 标注为下方的短线段。图中 (a). 蓝色线的重构效果: SNR = 31.02, Spike Ratio = 80% (b). 绿色线的重构效果: SNR = 36.68, Spike Ratio = 100%。底部竖线段表示检测出 spike 的对应位置。

可以看出，在右图中 $x=779$ 处检测到的 spike 并未在左图中检测出来。这是因为信号在这里压缩时有一部分被丢掉了，以至于在 spike 检测的时候没有达到式2.16中的阈值。注意高 SNR 经常伴随着高 Spike Ratio, 但也不一定总成立。因为 SNR 更多的反映了信号在低频能量上的特点。

3. Compression ratio 此外，我们还用压缩 (Compression Ratio, CR) 比来从数据冗余减少情况的角度度量压缩效果。压缩比定义为信号的压缩后大小除以原始信号所占空间。

在下面的实验中，我们在以上三个评价标准上来衡量压缩算法的有效性。

2.6.3 参数设置

本小节中，我们讨论如何选定三个参数: T_{LH} , 幅值过滤器的阈值; ω , 量化表的比例; 和 S_b , 预处理中每个数据块的大小。

在我们提出的压缩框架中，有两个步骤会造成信息丢失：一是符号编码中的均值代替策略，而是高幅值分量的量化。这两个步骤都是在频域操作的，但是符号编码环节中，信息丢失最大为相应的量化表的值；而量化部分信息丢失最大为量化表对应值的一半。具体用哪一种方法进行压缩取决于高低幅值分量之间的界限，也就是取决于幅值滤波器的阈值 T_{LH} 。随着 T_{LH} 的升高，会有更多分量被符号编码压缩，带来更大的信号损耗而提高压缩率。因此， T_{LH} 可以看做是重构效果和压缩比之间的协调系数。在我们的模拟中， T_{LH} 在整个数据集上测试，以求的最佳的压缩效果。

另一个重要参数是量化表。为了在压缩过程中节省参数，量化表以 LAC 分量的平均幅值作为结果，在符号编码中进行共享。因此，QT 会随着 T_{LH} 的增加而增加，从而在量化过程中可以粗化数据，降低其分辨率。这里我们想了解能否通过调整 QT 的比例达到更好的压缩效果。所以，我们用 QT 乘以一个比例系数 ω 来调整 QT 进行试验，其中， ω 在 $[0.5, 2.5]$ 之间，以 0.5 为步长进行测试。

在不同参数下，我们的实验结果如图2.10所示。每个子图的横轴都是阈值 T_{LH} 。不同大小的量化表用比例系数 ω 表示，其中 $\omega = 0.5$ (—), 1 (—), 1.5 (···), 2 (-.-.-) and 2.5 (-*-*). 每个子图的结果都是通过系统地调节阈值 T_{LH} 和比例系数 ω ，然后在整个数据集上做结果的平均而来的。保持 ω 不变，随着 T_{LH} 的上升，可以清楚地看到 SNR 和 Spike Ratio 都有所下降，这是因为高阈值会同时放大 QT，相反这样会带来更优的压缩率。从图中可以看出，从 SNR 和 Spike Ratio 的角度来看， $\omega = 1$ 的结果总是最优的；从 Compression Ratio 的角度来看， $\omega = 1$ 的配置排在第二。容易理解为什么 Compression Ratio 随着 ω 的增大而变化（降低），因为 ω 更意味着损失更多数据，导致 Compression Ratio 降下来。

图2.10 展现了 ω 是怎样影响压缩比和信号保真度的，但是由于没有一个 ω 在 3 个评价指标上都达到最佳结果，所以我们仍然很难评判哪个 ω 最好。为此，图2.11直观地表示出，在相同 Compression Ratio 的情况下，另外两个压缩评价标准的值。该图是图2.10的一个变换，图中横轴以压缩比作为自变量，纵轴 (a). SNR, (b). Spike Ratio 作为因变量，每条曲线表示在不同 QT 的比例系数和阈值下的压缩效果。可见，在相同 Compression Ratio 下 $\omega = 1$ 总是达到最好的效果。因此，对于无监督压缩，QT 的比例系数为 1 时可以权衡压缩

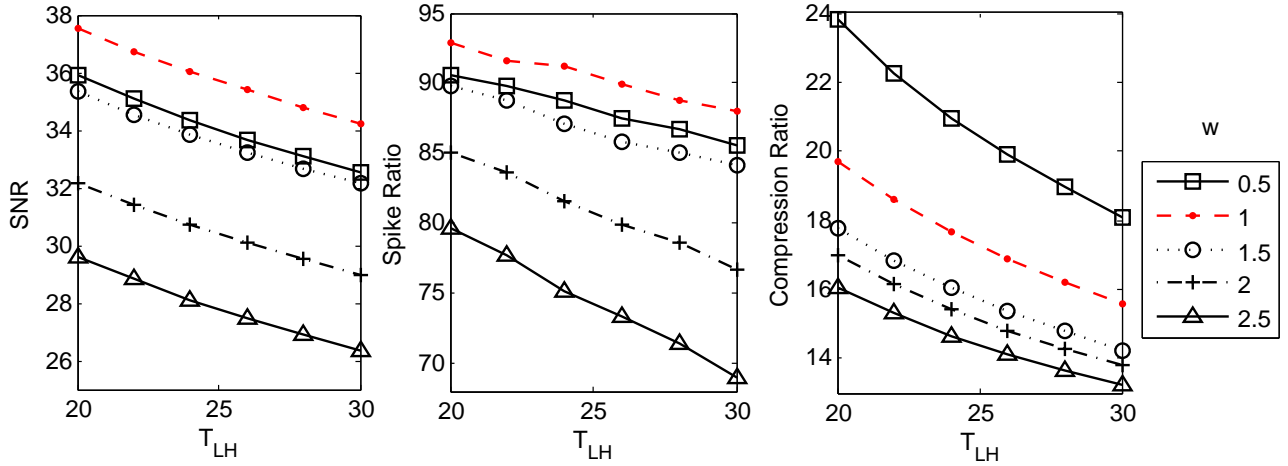


图 2.10 不同 T_{LH} 和 QT 下的 SNR, Spike Ratio 和 Compression Ratio。

率和重构效果，给出一个理想的压缩结果。

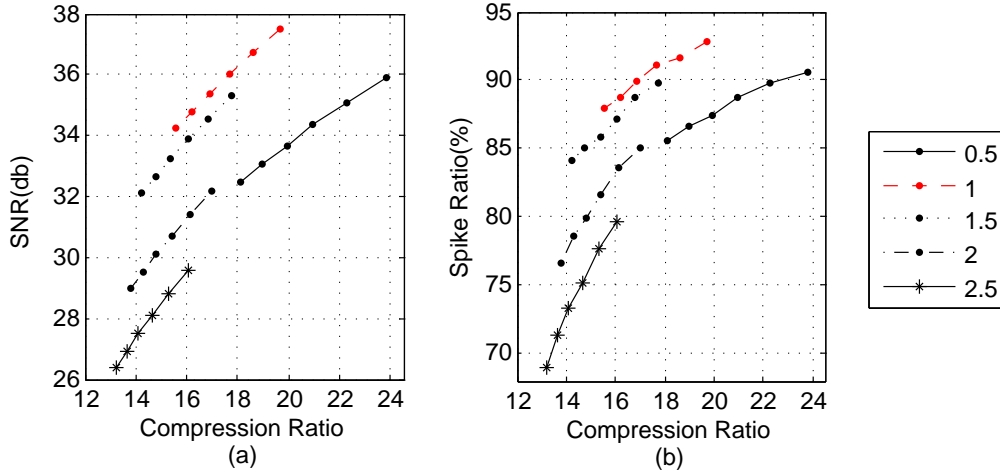


图 2.11 不同量化表比例系数下的压缩效果。

T_{LH} 的选择取决于我们的压缩要求。例如，如果希望 SNR 大于 30db 且 Spike Ratio 不小于 90% 的话，选择 $T_{LH} = 24$ ，可以得到平均 Compression Ratio 为 17.75%，SNR 达到 36.24db，Spike Ratio 大于 90%。

最后需要确定的参数是预处理中的数据块大小 S_b 了。在上述实验中，我们暂时都取 $S_b = 1600$ ，这里我们来讨论能否通过更改 S_b 达到更好的压缩效果。实验结果如图2.12所示，其中横轴表示块大小 S_b ，固定 $T_{LH} = 24$ ， $\omega=1$ ， S_b 在 1500 到 28500 之间进行测试。

该图说明随着 S_b 的增长，信号的保真度先提高后下降。原因如下：首先，更大的 S_b 会使 DCT 系数更为精确，于是在 IDCT (inverse DCT) 的过程中降低误差。但是，这样会生成很多低幅值的 DCT 系数，这样会使得归入 LAC 分量的值增多，也就是，会有更多分量选用符号编码方法进行压缩，这使得 loss 增大。因此，压缩保真度先增后减反映了上述

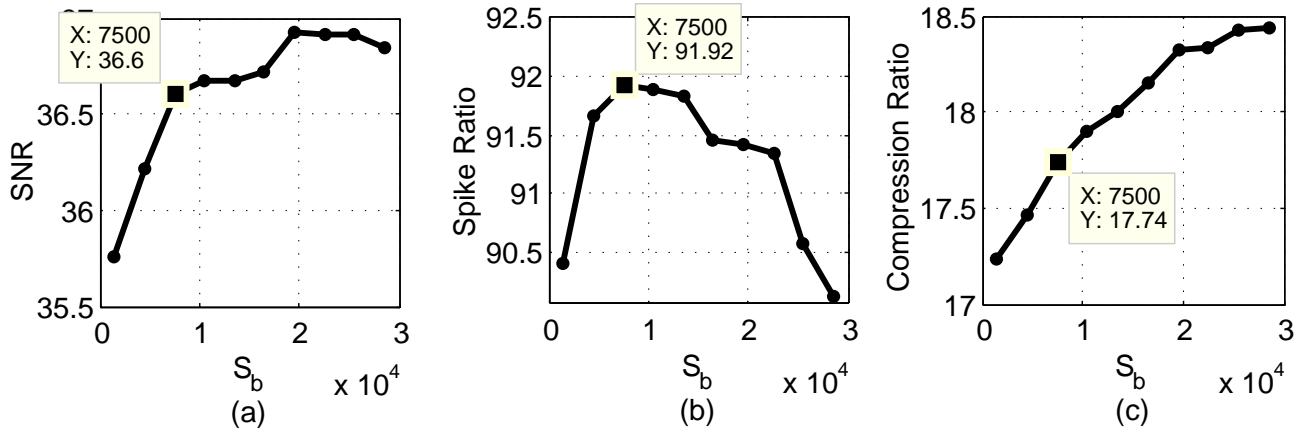


图 2.12 不同块大小 S_b 下进行数据分割的压缩预处理的压缩结果。

两个因素之间的平衡。根据给定的实验结果，可以将数据块最佳大小设置为 $S_b = 7500$ ，这时达到整个数据集上平均 Compression Ratio 为 17.7%, SNR 为 36.6db, Spike Ratio 为 91.9%。

2.6.4 符号编码有效性

在第2.4.1节中，我们提出了符号编码方法来压缩低幅值的 DCT 系数。本节中给出实验结果，探讨符号编码的有用性。在上一节中我们已经选出最佳参数。给定这些参数设置，我们设计一个对比试验，保持高幅值部分压缩方法不变，对低幅值部分不采用符号编码而是整个丢弃，以此来看符号编码所带来的改善。比较结果如表格2.1所示。

符号编码	SNR(db)	SR	CR
有	31.4	83.7%	13.1%
无	36.6	91.9%	17.7%

表 2.1 SNR, Spike Ratio and Compression Ratio With and Without Symbol Encoding (Low-Amplitude)

可以清楚地看出，将低幅值分量考虑进来可以保存很多有效信息。虽然这样会升高 Compression Ratio，但从 SNR 和 Spike Ratio 的角度来看，通过符号编码后重构效果有了很大提升。这是由高频部分系数的内部分布决定的。因此，高频部分的符号编码是一个有效的压缩方法。

2.6.5 方法比较

由于在神经电信号压缩方面缺少统一的压缩标准作参照，而音频也是单维度时序信号，所以我们在本节中选择与 state-of-art 的音频压缩算法作比较。同时，我们也与其他传统数据压缩方法作比较。通过与有损和无损压缩算法比较的试验结果可得，我们的压缩算法能够权衡重构效果和 Compression Ratio。

2.6.5.1 无损压缩

本小节中讨论无损音频压缩和通用数据的压缩技术。无损压缩方法通过更为紧凑的编码格式对原文件进行编码，使得数据编码后解压所得文件与原文件完全一样。对音频压缩，FLAC 之类的编码方法利用线性预测来估计信号频谱，可以对通用波形的 Compression Ratio 达到 50% 到 60%^[26]。但是神经信号不同于音频信号，神经信号更为复杂，难以预测。所以这类压缩方法即使用到神经信号压缩，也不能获得较好的 Compression Ratio。类似的，通用数据文件压缩方法，如 Zip, 7-Zip 和 RAR 也不能达到相对较低的压缩率。表?? 显示出不同无损压缩技术的压缩比。可见，在神经信号上最好的压缩方法是 APE（Monkeys Audio），得到最小压缩率为 56.88%。注意，尽管神经信号被这些方法压缩后不能得到很好的压缩率，但文件可以被完好的重建出来，因此，SNR 是无穷大。

表 2.2 无损压缩方法性能比较

	无损压缩编码					Ours
类型	音频编码			文档文件格式		SNR=36db
	Lossless WMA	FLAC	APE	Zip	RAR	Spike Ratio=92%
压缩比	70.89%	54.27%	53.08%	70.04%	60.91%	17.74%

2.6.5.2 有损压缩

不同于无损压缩方法，有损音频压缩利用了人类听觉感应特点，即只对特定频率和幅值信号敏感，而丢弃其他对声音辨别率影响较小的琐碎信号。所以音频的有损压缩只专注于量化病变吗那些容易感觉到差异的频谱部分。本小节中以一个 state-of-art 的音频压缩方法（Advanced Audio Coding）为例，与我们提出的压缩算法进行比较。Advanced Audio Coding（AAC）是 MPEG-2 标准的一部分，与 MP3 相比，AAC 可以提供更好的信号质量，同时将信号 Compression Ratio 多降低 30%。图 2.13 分别显示出这两种方法的 Compression ratio 对应重构效果。

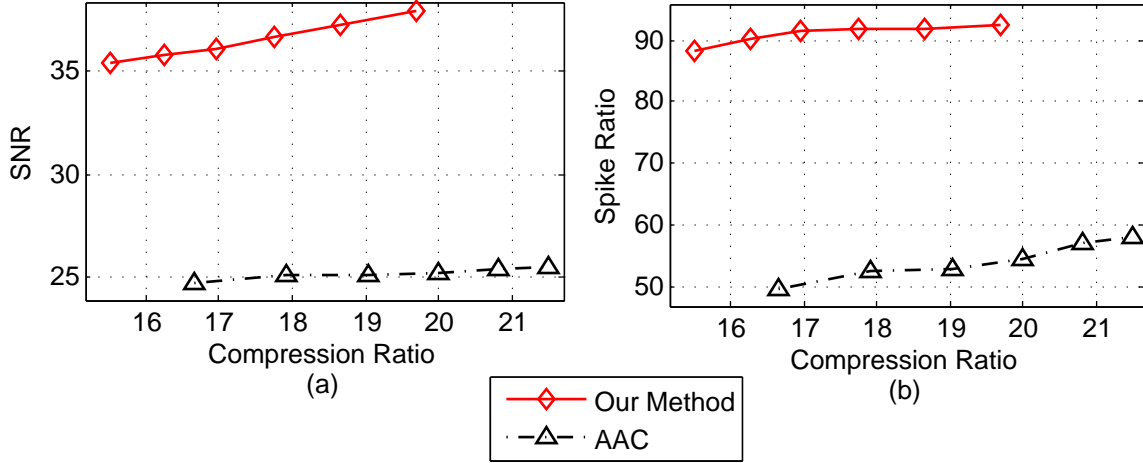


图 2.13 我们方法与音频编码方法压缩效果比较。

为了达到好的重构效果，我们对音频压缩选择高比特率，从 300kbps 到 600kbps。

但是，对于神经电信号，这两种方法的压缩效果并不理想。图中显示，在相同的压缩比下，我们的方法在两个信号保真度评判标准上都比 AAC 高，SNR 超出 AAC 46.4%，Spike ratio 超出 AAC 80.4%。原因是音频压缩方法 AAC 在压缩过程中不关心哪部分对神经信号的处理比较重要。实验结果也说明了神经电信号的信号特性有别于音频信号，所以常规压缩方法不能很好地作用于神经电信号。

2.6.6 计算代价

之前的小节描述了我们提出的算法在权衡 Compression Ratio 和重构误差时的有效性。最后，我们讨论该算法的压缩效率。一下结果是从我们在 MATLAB 上的实现中得来的统计结果。初始化过程获取量化表，哈夫曼码表和无损压缩部分的分界线 B 速度为 2.86Mb/s，压缩过程为 0.13Mb/s，解压速度为 0.14Mb/s。如果用其他语言实现该算法压缩过程应该会得到更大提速。

2.7 结论

本章提出了一个运动皮层神经电信号的压缩方法。它在信号频谱采用提出的双阶编码方法进行信号压缩。时序信号首先转换到频域，然后通过一个幅值滤波器将信号分为高幅值分量和低幅值分量，然后分别对其进行编码。为了压缩低幅值分量，我们采用符号编码，对每个值只记录 1bit 的符号；对于高幅值分量，我们将其量化，然后采用混合编码方法进行编码。

之后，我们将所提出的方法与其他压缩方法进行了比较。可以看出，在一定的 Compression Ratio 的情况下，与其他方法相比，我们的方法在重构效果（SNR 和 Spike Ratio）上都更胜一筹。最终我们的压缩算法达到 17.7% 的压缩比，SNR 为 36.6dB，并保证 91.9% 的 Spike 保真度。该结果与其他神经信号压缩方法相比也很出众，其他神经信号压缩方法只能使 SNR 达到 15-26dB，压缩到元数据的 1-20%^[27-29]。我们的算法在运动皮层（M1 区）采集的神经电信号上进行了验证，但是它也可以应用到其他神经信号上。而且，如果考虑到其他信号的通道间相关性，可以进一步提升压缩性能。

而且，基于统计结果而得的量化表，哈夫曼码表和其他参数可以根据信号被预先计算好。我们的压缩框架原型成功地在猕猴运动皮层捕获信号上通过测试。压缩效果比较理想，只是有个缺陷，本算法需要迭代求解，因此比较耗时，但由于压缩过程没有时序要求，所以可在后期通过并行加速。

第3章 卷积神经网络分析信号

3.1 深度学习介绍

3.1.1 传统机器学习方法与局限性

在之前的 50 年左右，传统的模式识别模型用手工定义的特征进行特征提取，通过对数据的分析选取可训分类器进行模型构建。最近 10 年，借助现代计算机计算能力的提高和大数据量的爆发，神经网络方法得以重新广泛应用，在很多领域都达到非常好的效果，我们称这种利用大规模网络进行模式识别方法为深度学习（Deep Learning），也叫 End-To-End Learning。不同于传统模型采用固定特征，或者固定 kernel（核函数）进行样本度量；深度学习采用可训特征（或可训的 kernel），然后将特征作为可训练的分类器输入，进行训练，如表3.1。

	特征	分类器	特点
传统方法	人工定义的特征	简单可训练分类器	特征设计费时，需强业务背景
深度学习	训练特征提取模型	复杂可训练分类器	End-to-End learning, feature 易操作

表 3.1 深度学习与传统模式识别方法

历史上，第一个有学习功能的机器为 1960 年提出的 Perceptron^[30]，也是神经网络的一个基本单元。Perceptron 是一个简单特征提取器上加载的一个线性分类器：

$$y = \text{sign}\left(\sum_{i=1}^N w_i F_i(x) + b\right) \quad \text{公式 (3.1)}$$

其中 x 为数据， $F_i(x)$ 为 x 的第 i 个特征， w_i 为相应的特征权重参数， b 为常参数， sign 为分类器的非线性函数，对于二类分类， sign 函数将结果映射到 (0,1)。

目前最普及的实际应用也用到了线性分类器的一些变种，或者叫模板匹配（template matching）。但是由于其底层的特征提取器需要反映特定信号的特点，所以需要由特定领域专家来设置。比如图像处理领域，对于不同任务（图像分类，图像分割，图像跟踪等），所要求的特征就各不相同，需要针对特定任务定义图像特征。此外，传统方法也很难设计 kernel，从而不容易表达对距离的度量，仍然以图像来说，最简单的距离度量思路是对应像素相减，但是这显然不能表达图像语义层的相似信息。

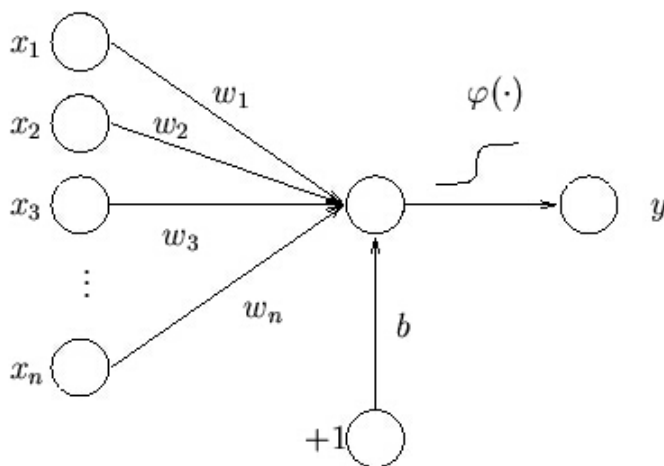


图 3.1 Perceptron 图例

为了能使特征更加灵活而且不过多依赖于专家指定的特征，很多方法提出，可以先人为定义简单特征，然后通过无监督学习方法进一步得到中间层特征，将其输入分类器进行最后分类。典型的无监督特征学习方法如混合高斯模型^[31-38]，K-Means^[39-43]，和 Sparse Coding^[44-49]。但这仍然不能解决以下几个问题：

1. 建立传统模型代价大

每个领域，每个任务都需要设计特征。即便有了中间特征层进行无监督的特征选择，庞大冗余的基础特征的设计也是耗时费力的。随着工业界对不同任务的需求，需要建立很多基础特征、模型，代价也很大。

2. 无法很好地利用计算性能

计算机性能的大幅提升本可以用来帮助加速机器学习，但传统机器学习需要人工定义模型，从而使模型规模受限，不能很好地利用计算性能和大量数据。

3. 人工定义特征效果欠佳

目前，自动学习的特征已经在图像、语音等很多领域强于人工定义的特征。而且如果需要增加特征维度进行大规模学习就需要再手工定义更多特征，而不能简单地够自动按比例扩大。

3.1.2 深度学习方法介绍

深度学习是近几年来很热的机器学习算法，在语音识别，计算机视觉，自然语言处理等领域打破了维持了多年的竞赛记录。以最典型的时序模型——语音识别的发展为例，在

1980 年代早期, 主要应用 Dynamic time Warping (DTW)^[50-53], 输入底层特征, 通过无监督学习所得中间层特征输入分类器进行分类。1985 年后,^[54-56] 提出在中间层用隐马尔科夫模型描述一个序列出现的概率, 得到进一步改进。2010 年左右, 使用深度神经网络进行逐层有监督学习达到最佳效果^[57;58]。

同传统方法的基本方法类似, 深度学习也是从数据分别生成底层特征, 中间层特征, 最后加入高层特征, 输入分类器进行分类, 即学习数据的结构化表示。其形式化表示为:

$$y = f(W^k f(W^{k-1} f(\dots f(W^0 X) \dots))) \quad \text{公式 (3.2)}$$

其中, W 为权重参数, k 为层数, $W^i X$ 为输入 X 的线性表示, f 为非线性函数, 如 \tanh 函数。深度学习就是优化 y 与 groundtruth 之间的距离, 使之最小化。从图像角度, 如图3.2所示^[59;60], 图中 (a), (b), (c) 分别为自动学得的底层特征, 中层特征和高层特征, 其中底层特征学习图像的浅层特征, 在所有类别中共享, 如 (a) 中, 学到的特征类似 Gabor 滤波器所提取的边缘特征^[61;62], 从中间层到高层依次学习图像更深层的语义特征, 如有语义的显著图像区域, 高层特征更为稳定, 也具有类属性。从自然语言处理的角度, 初始输入为字符, 从底层向上依次学习单词, 短语, 长句, 文章。

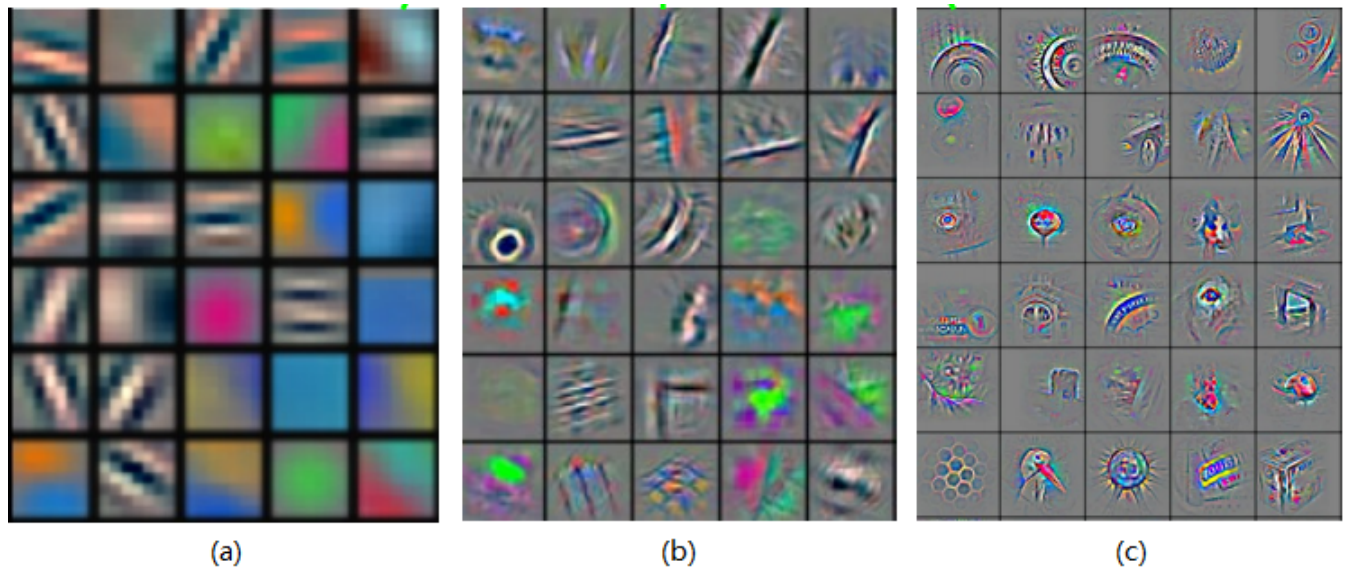


图 3.2 各层次 feature

除了自动学习特征外, 深度学习和传统模型还有一个重要区别就是“深”。一般来说, 深度结构由多层含参的非线性模型组成, 以形成特征的层次结构。在^[63;64] 中讨论了深度模型的必要性。浅层结构以现在的 kernel machine^[65] 为例, 如 Support Vector Machine(SVMs)^[66;67]。这些方法定义特征为一系列 kernel 函数连接而成的向量, 在训练数据中进行模板匹配, 如式3.3所示。

$$\varphi(x) = [k(x, \mu_1), k(x, \mu_2), \dots, k(x, \mu_n)] \quad \text{公式 (3.3)}$$

式中, μ_i 为数据样本的一部分, 即模板样本; $\varphi(x)$ 为样本 x 的特征。而后 $\varphi(x)$ 通过线性组合进行分类:

$$F(x) = W^T \varphi(x) \quad \text{公式 (3.4)}$$

可见, kernel 方法就是一个简单的模板匹配层连接一个线性函数层, 由于方法中模板都是从原始训练数据中提取而得, 所以 kernel machine 的第一层可视作一种简单的无监督方法, 只有式3.4中参数 W 的学习为有监督部分, 没有涉及特征的层次结构, 因此不是深度模型。同样, 只有一个隐层的模型 (如 multilayer perceptron) 也不算深度模型。又如分类回归树 (Classification and Regression Tree, CART), 其中所有决策都是在输入空间定义的, 同样没有根据特征的层次结构进行学习, 因此也不属于深度模型。虽然有一些理论保证浅层结构可以以任意精度拟合任何复杂函数^[68-70], 它们却无法保证特征的高效表征。而深度模型可以从根本上通过简单设置网络结构更有效地表示特定函数。最典型的深度结构就是有多个隐层的神经网络, 其中每个隐层节点之间的连接权重 (如图3.1中的 w_i) 都是通过有监督学习而来的, 因此可以很好地表征数据属性。

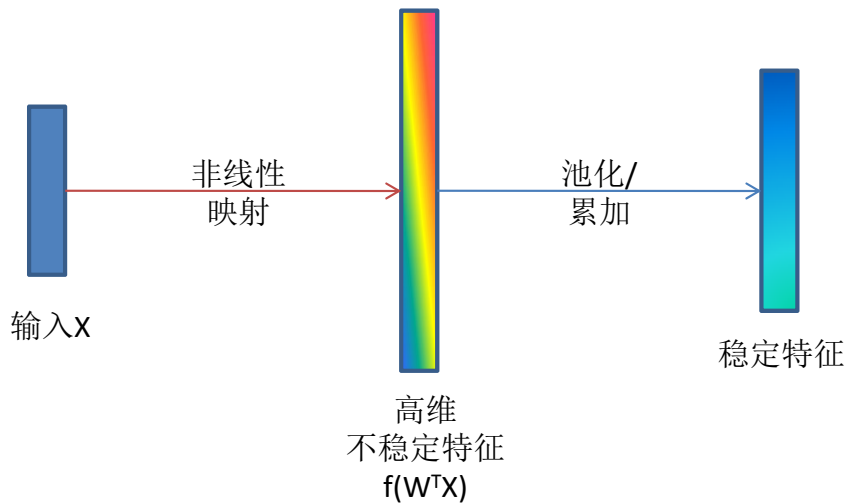


图 3.3 稳定特征生成方法

神经网络在 1940 年代就开始了相关研究^[71;72], 但早期提出的神经网络多为线性回归的变种, 没有很大突破。1965 年,^[7] 提出了第一个深度学习模型——Group Method of Data Handling (GMDH), 它通过回归方法进行网络训练, 再通过决策正则化去掉多余节点。直到现在仍有很多应用以 GMDH 为基本框架, 如^[73-78]。1960 年代, 受神经科学的启发, 神

神经网络也随之发展。当时神经科学在猫的视觉皮层发现了简单细胞和复杂细胞^[7]，简单细胞对局部视觉特征（如边缘的方向）敏感，会做出一定响应；而复杂细胞相对简单细胞表现出更为鲁棒的响应，即不受空间变化的影响，可以在视网膜一块邻域区域中选出简单细胞的一个输入作为输出。受此启发，1979年^[79]第一次在 Neocognitron 中引入了类似卷积神经网络的概念。Neocognitron 是为手写体识别提出的一个有层级结构的多层神经网络，它不仅可以识别训练样本中的模式，还可以识别训练样本经过平移，旋转或其他变换后的模式。其结构和现在我们用到的前向卷积神经网络比较相似，层间连接也应用了现在我们常用卷积神经网络类似的部分权值共享（详见第3.2节）。只是，Neocognitron 通过无监督学习，根据样本获得不同的 pattern，而不同于现在的有监督误差反传方法。此外还有一些小差别，比如在 pooling 策略上，Neocognitron 采用空间域平均，而且虽然 Neocognitron 的层级较深，但是它并没有在效果中考虑到深层带来的效果。

由于 90 年代计算资源和数据量受限，当隐层层数大于 2 的时候深度网络训练困难^[80]，所以之前神经网络的研究工作主要集中于浅层结构。随着现在数据量的爆发，神经网络的研究变得重新热门起来。其基本思想是学习具有不变性的特征，即，将输入数据映射到一个非线性的高维空间，使数据变得可分。如3.3所示，输入经过非线性函数的映射到高维特征，使数据变得可分，但是这样的高维特征可能由于训练数据间差异导致不稳定，所以将高维数据中相似语义信息的成分通过池化（pooling）或累加等方法整合到一起，形成稳定特征。

3.1.3 深度神经网络结构与训练

3.1.3.1 网络结构

从网络结构来分，神经网络有前向网络（包括 multilayer neural nets^[68;81;82] 和 convolutional nets^[83;84]），回馈网络（包括 stacked sparse coding^[85]，deconvolutional nets^[86]）和双向网络（包括 deep boltzmann machines^[87-89]，stacked auto-encoder^[90-92]）三类^[93]。目前，前向网络使用最广，如图3.4所示为一个前向网络的示意图。

其中， x_i 表示第 i 层的输入，整个网络的输入为 x_1 ； w_i 表示第 i 层的参数， $w_i^T x_i$ 得到 x_i 的线性组合； f 为非线性函数，每一层通过非线性函数将输入数据映射到下一层，如图中蓝色箭头所示。经过每一层非线性映射 f_i 的刀下一层输入，即

$$x_i = f_{i-1}(w_{i-1}^T x_{i-1}) \quad \text{公式 (3.5)}$$

给定 x_1 对应的样本 label y ，我们的目标函数为 $L(f_n(x_n), y)$ ，其中 L 为损失函数。常见的损失函数有平方损失，指数损失，对数损失等。

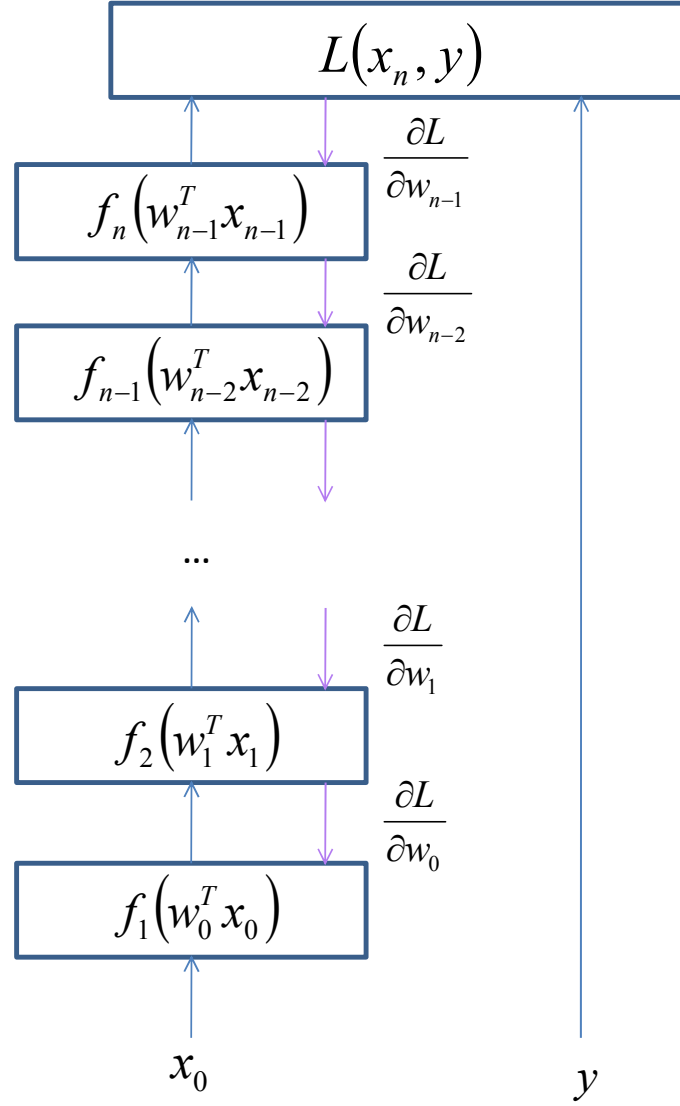


图 3.4 前向网络的网络结构

3.1.3.2 网络训练与误差反传

自从 1960 年以来, 不断有文章^[94? -99] 从变分法中的欧拉-拉格朗日方程出发讨论如何利用梯度下降 (gradient descent, 也叫 steepest descent) 的方法^[100] 来进行神经网络中多层非线性可导函数的优化。在这样的神经网络中, 可以通过链式法迭代地则对每个神经元求导^[97]。之后, 1970 年, 误差反传算法 (back-propagation algorithm, BP)^[69;101] 首次在 Linnainmaa 的硕士毕业论文^[102] 中提出, 其中描述, 误差反传算法可以高效应用于任意神经网络结构。此后 BP 迅速应用到神经网络的训练中, 并实现了给定可导函数自动求解导数和 BP^[103]。其基本思想如下: 由图3.4和式3.2, 在前向算法中从输入到输出经过了一系列非线性可导函数 f 的映射。每次针对一个样本, 计算出其输出和样本真实 label 之间的 loss L , 然后自上而下逐层更新权重参数 w (如图中向下的紫色箭头所示), 使得当前误差

可以减小。用所有样本依次对权重做完更新后叫做一轮迭代。不断做 n 轮迭代，直到全局 loss 降到某个预设阈值，这时网络就已经学好了。其中，对 w_i 更新权重时采用梯度下降的方法：

$$\Delta w_i = -\gamma \frac{\partial L}{\partial w_i} \quad \text{公式 (3.6)}$$

$$w_i = w_i + \Delta w_i \quad \text{公式 (3.7)}$$

式3.6中， γ 表示学习率，就是每一步朝梯度下降方向走的步长，是一个正的常数。注意这里的负号表示 loss 是朝梯度下降的方向走。式3.7 表示 w_i 每一步的更新。但是 L 的表达式中难求对每个 w_i 的导数，这里我们用 net_j 表示第 j 个隐层节点，有

$$net_j = \sum_i w_{ij} o_i \quad \text{公式 (3.8)}$$

即上一层输出到该节点的线性组合，其中 o_i 表示节点 i 的输出， w_{ij} 表示节点 i 到 j 之间的连接，有 $o_j = f(net_j)$ ，则 L 对 w_{ij} 的导数如式3.9。

$$\begin{aligned} \frac{\partial L}{\partial w_{ij}} &= \frac{\partial L}{\partial net_j} \cdot \frac{\partial net_j}{\partial w_{ij}} \\ &= \frac{\partial L}{\partial net_j} \cdot \frac{\partial \sum_k w_{kj} o_k}{\partial w_{ij}} \\ &= \delta_j \cdot o_i \end{aligned} \quad \text{公式 (3.9)}$$

其中 o_i 为前向过程中第 i 层的输出， $\delta_j = \frac{\partial L}{\partial net_j}$ 为第 j 层中从后向前传递的错误，我们称之为反向传播误差 (backpropagated error)。由公式3.6, 3.7和3.9可知， i, j 节点之间的权重的参数 w_{ij} 更新由节点 j 在输出端的反向传播误差乘以节点 i 的输出 o_i (也就是节点 j 的输入端) 而得，如图3.5 所示，黑色线条表示前向通路，红色线条表示反向传回的误差 Δw_{ij} 。

对于多层网络，考虑如何计算反向传播误差 δ_i 。对于整个网络的输出单元 net_n ，可以直接算得 $\delta_n = \frac{\partial L}{\partial net_n}$ ，例如对于 L 取 square loss 的情况，有

$$\begin{aligned} L(y, net_n) &= \frac{1}{2} (net_n - y)^2 \\ \delta_n &= \frac{\partial L}{\partial net_n} = net_n - y \end{aligned} \quad \text{公式 (3.10)}$$

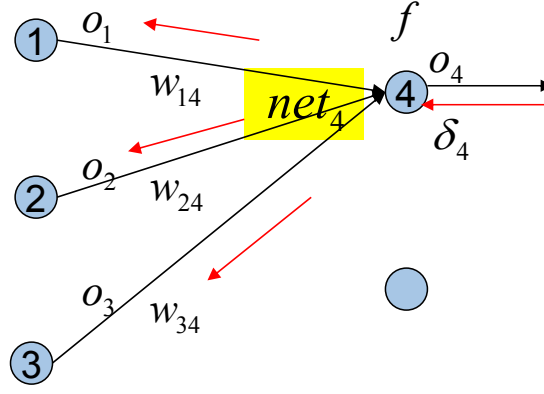
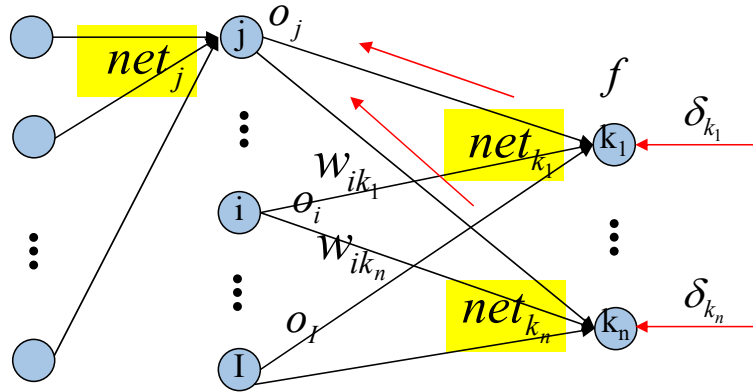


图 3.5 误差反向传播示意图

对于隐层节点 net_j ，通过链式法则求导可得，如图3.6和公式3.11所示。

$$\begin{aligned}
 \delta_j &= \frac{\partial L}{\partial net_j} \\
 &= \sum_k \frac{\partial L}{\partial net_k} \cdot \frac{\partial net_k}{\partial net_j} \\
 &= \sum_k \delta_k \cdot \frac{\partial \sum_i w_{ik} o_i}{\partial net_j} \\
 &= \sum_k \delta_k \cdot \frac{\partial \sum_i w_{ik} f(net_i)}{\partial net_j} \\
 &= \sum_k \delta_k \cdot w_{jk} f'(net_j)
 \end{aligned}
 \tag{公式 (3.11)}$$

图 3.6 $\frac{\partial L}{\partial net_j}$ 链式法则示意图

这样就可通过将误差逐层传递训练网络了。整个算法流程如算法3所示。在每一次迭代中，首先前向计算出每个节点的输入 net_j 和输出 o_j （见 3 6 行）。然后用误差反传算法，在输出层直接用损失函数对节点求导计算反传误差（9-11 行），其他隐层节点用公式3.11计算反传误差（12-14 行），最后更新权重参数（15-18 行）。

Algorithm 3 前向网络的误差反传算法

Input: Data x , Label y **Output:** updated weight w

```

1: for  $t=1\dots T$  do % 迭代次数
2:   % forward propagate
3:   for all unit  $j$  not  $\in$  input layer do
4:      $net_j \leftarrow \sum_i w_{ij} o_i$ 
5:      $o_j \leftarrow f(net_j)$ 
6:   end for
7:
8:   % back propagate error
9:   for all unit  $k \in$  output layer do
10:     $\delta_k \leftarrow \frac{\partial L}{\partial net_k}$  % 直接求导计算
11:   end for
12:   for all unit  $j \in$  hidden units do
13:     $\delta_j \leftarrow f'(net_j) \sum_k w_{kj} \delta_k$  % 根据公式3.11
14:   end for
15:   for all weight  $w_{ij}$  do % 根据公式3.7和3.9更新权重参数
16:     $\Delta w_{ij} \leftarrow \frac{\partial L}{\partial w_{ij}} = \delta_j \cdot o_i$ 
17:     $w_{ij} \leftarrow w_{ij} + \Delta w_{ij}$ 
18:   end for
19: end for

```

BP 尽管在理论上支持深度网络的训练,但是实际应用中 BP 一般只对较浅的模型有效^[104],因此提出了很多 BP 的改进算法,如:

1. 关于 BP 加速

为了加速 BP, Rumelhart 等人在 1986 年引入动量 (momentum)^[105]; Fahlman 向线性的激活函数 f 的斜率加入 ad-hoc 常数^[106]; Sperduti 不但对 x 进行梯度下降,同时对梯度参数也进行梯度下降;^[107] 将不同的 bp 加速策略做了比较。

2. 关于 BP 学习率

一般地, BP 有两种学习率设定方法: 全局学习率^[108-110] 和对每个权重都设一个自己的学习率^[111;112]。在线学习中, $\text{vario} - \eta$ ^[113] 算法设置学习率与当前节点梯度的标准差成反比,从而可以不受随机权重初始化的影响。

3. 如何增强模型普适能力 (generalization)

为了提高模型的 generalization, 一些方法采用 weight decay^[114;115], 它本质上是一种正则化方法,通过在 loss function 中加入权重参数的平方项来鼓励更小的权重,惩罚大权重。在线性模型中,这种方法就是 ridge regression。加入 weight decay 一来可以限制模型的复杂度,防止模型过拟合,二来如果激活函数不能将输出规范化到输出应在的范围内,大权重很可能让输出远远大于数据范围。此外,大权重还会陷入 Bias/Variance Dilemma^[116],导致输出数据波动很大。Hinton 等人^[117] 从贝叶斯角度解释了 weight decay 可以从 weight 的高斯先验假设中得来,证明了这不是一个简单的猜想。另一些方法通过在 weight 中加其他先验实现同样的功能^[118-121]。

总结一下,误差反传算法通过求损失函数对每个权重参数的导数对它们进行更新,更新过程从输出端到输入端依次向下传递误差。与数值差分相比,误差反传方法更适应计算机运算,而且计算更为高效。在实际应用中, BP 也有一些条件,当问题定义的复杂度很高,不能写出解析解,但是确定有解的时候我们采用 BP 进行网络优化。否则可能用传统方法或者查找表方法会更为简单精确,具体问题具体分析。

在第3.1.3.1节中这三种为网络结构中,卷积神经网络属于广泛应用的前向神经网络。我们将在第3.2节中介绍卷积神经网络的结构与训练过程,在第3.3节中介绍我们的数据与网络配置,最后在第3.4中给出实验结果。

3.2 卷积神经网络

在上一节中我们已经提到，卷积神经网络（Convolutional Neural Networks, CNN）是前向网络的一种，其原型始于1979年Fukushima提出的Neurocognitron^[79]，但在后期经过了很多改进。1989年，Lecun将误差反传算法应用于带有共享权值（详见第3.2.1节）的卷积神经网络进行训练^[122]，这一工作的内容是现在我们前向神经网络的重要部分。在20世纪90年代，CNN在指纹识别^[123]、人脸识别^[124]、语音识别^[125]、物体识别^[126]、文档识别^[1]上的应用相继出现并予以商用。

近几年来，随着计算机计算能力的增强和数据量的激增，卷积神经网络在各个领域的应用更为广泛，也得到了更好的效果。以机器视觉领域为例，09年，Turaga（MIT）等人用学习的方法，以卷积神经网络为模型训练生成相似图^[127]，使得和传统人工定义分割函数的方法相比，分割精度有了很大提升。2011年，Sermanet（NYU）和Lecun将CNN应用于交通标志分类^[128]，在GTSRB竞赛数据集^[129]上达到99.17%的最高准确率，超过人的识别能力。对于图像分割任务，很多算法需要首先生成一张相似图（affinity graph）然后在其上做分割。2012年，Alex Krizhevsky（Google）在ImageNet^[130]上应用深度卷积神经网络进行图像分类^[84]。该任务要将130万高清图像分成1000类，Krizhevsky采用5层卷积层，共50万个神经元节点，6000万个参数进行训练，达到了当时图像分类的最佳效果。2014年，Taigman（Facebook）等人应用卷积神经网络建立了DeepFace^[131]，在包含4000多人的4百万人脸图像数据集上进行训练，在LFW数据集^[132]上进行人脸识别，结果达到97.35%的识别率，近乎和人的识别水平相当。整个过程为检测人脸->对齐->3d人脸表征->分类，其中分类用的是多达1.2亿个参数的9层非标准卷积神经网络。同年，Simonyan（Oxford）等人将深度卷积神经网络应用到视频进行行为识别^[133]。他们提出了一个双CNN框架，分别在时间和空间方面各应用一个CNN，并证明再多帧的深度光流上应用ConvNet可以在有限数据集上达到很好的效果。Toshev和Szegedy（Google）用深度神经网络进行人体姿态估计^[134]。该方法在CNN上实现一个回归问题，回归目标为人体节点（如关节处），然后将多个这样的深度神经网络级联，在姿态估计方面达到了目前最佳的准确率。

此外，DeepMind团队（Google）^[135]在卷积神经网络上进行增强学习，实现了机器自动打游戏。Ciresan^[2]将深度神经网络应用于医学图像，进行癌细胞检测。自然语言处理方面，Denil（Oxford）等人通过将文档嵌入一个低维向量空间来提取文档语义信息并做归类，该模型基于卷积神经网络，在语义级别和文档级别同时学习卷积核。在框架方面，2014年，Goodfellow在神经网络中提出了一个估计生成模型的框架^[136]，通过同时学习两个相互对立的模型，一个生成模型G拟合数据分布，一个判别模型D估计样本从训练数据得到（而

不是从 G 得到) 的概率。最终目标最大化 D 犯错误的概率 (即让 G 尽可能地模拟原始数据分布), 其中 G 和 D 都是多层 perceptron。神经科学方面, Cadieu (MIT) 将卷积神经网络应用下颞叶皮层信号的建模^[137], 揭示了下颞叶皮层在基于视觉的物体识别任务上对信息的表征能力。也有工作将卷积神经网络应用于功能性磁共振成像 (fMRI) 数据大脑神经信号解码^[138], 结果和其他模式分析技术相比有所提高。Barnickel 等人基于卷积神经网络提出了 SENNA, 一个快速精确地从生物类文献中提取分析予以相关性的神经网络^[139]。

在第1.3节中我们已经介绍过 P300 神经信号的产生及其对应的 P300 检测任务。受以上文章启发, 本文中采用卷积神经网络为 P300 信号建模, 并应用于 P300 信号检测。下面, 我们先来看一下卷积神经网络是如何工作的。

3.2.1 CNN 网络结构

1. 卷积层

如图3.4所示, 卷积神经网络也是一个层级结构的前向神经网络。除了输入层和输出层, 中间的隐层的目标是从原始数据学得一系列非线性组合, 使得可以很好地结合当前任务表征信号。以图像为例, 输入为像素级数据, 每个像素本身不含有太多信息, 但是组合起来所得到的更高级别的特征可以表示物体类别等信息。尽管我们的神经信号不同与图像信号, 但是文本分析^[140], 语音识别等其他任务都受了图像的启发, 因其方便可视化, 所以我们采用图像进行解释。

以输入层和第一层隐层中的一个神经元节点连接为例, 如图3.7所示。图中黄色区域表示输入, 为 $32 \times 32 \times 3$ 大小的数据, 比如可表示一张深度为 3 (3 通道), 宽、高均为 32 的图像。图中的蓝色圆点表示与输入层相连的一个隐层节点, 连接项由权重参数组成。和大脑中视觉皮层的接收域 (receptive fields) 一样, 我们在卷积神经网络的隐层中也设置接收域, 该接收域可以视作一个在输入中进行扫描的线性滤波器, 为了减少参数数量, 该滤波器中的权重参数在输入层中共享。直观地想, 共享也是有原因的。因为该接收域的目的是发现输入空间中的一些特定模式, 这是不需要受到检测位置在图像中位置的限制的, 这也解释了为什么 CNN 网络学出的特征具有平移不变性。换句话说, 其目的是使我们要找的特定模式与其在输入数据的位置无关。卷积神经网络中, 我们称该接收域为核 (kernel)。例如图中为 5×5 的 kernel, 深度为 3 (和通道数相同)。由于 CNN 的连接层中共享权值, 所以该节点与输入层相连接的参数个数总共只有 $5 \times 5 \times 3$ 个。在输入层中, 从原点 (左上角) 开始分别向右方和下方扫描, 假如每隔 1 个像素进行扫描, 每个 $5 \times 5 \times 3$ 的区域块经 $5 \times 5 \times 3$ 的 kernel 卷积得到一个标

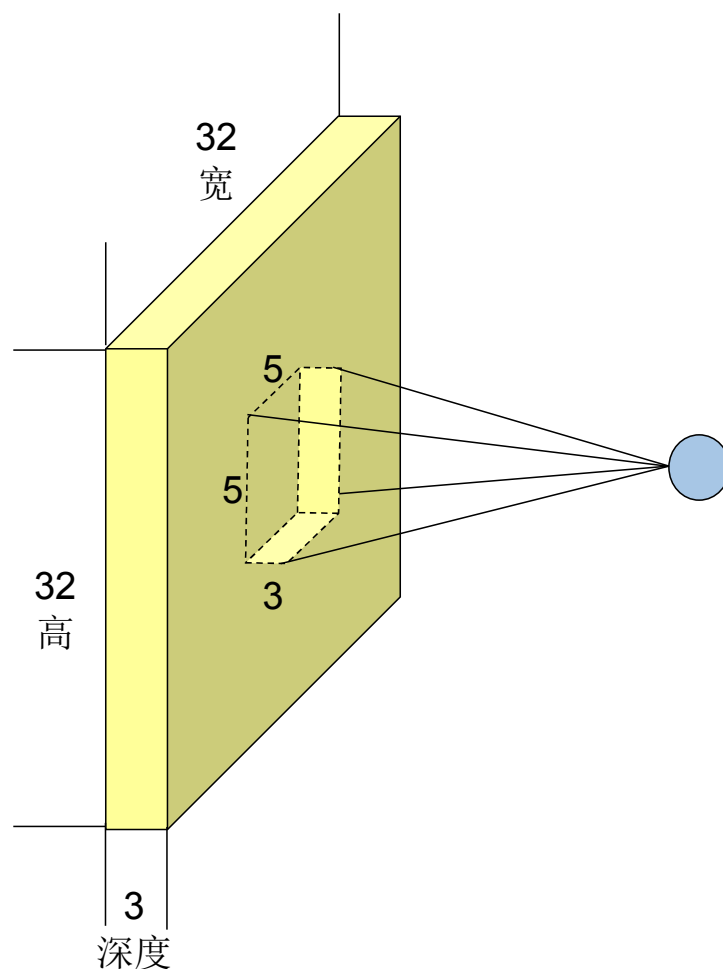


图 3.7 局部连接示意图

量，那么一个数据 $32 \times 32 \times 3$ 的数据经过卷积也就可以得到一个 $(32-5+1) \times (32-5+1)$ 大小的数据。

对于隐层中有多个节点的情况，我们在隐层的深度维进行扩展。假如我们希望隐层有 N 个节点，就令隐层的深度为 N ，那么隐层大小即为 $(32-5+1) \times (32-5+1) \times N$ 。刚才我们对于输入层每隔一个像素进行一次卷积，但是为了减小计算复杂度，我们还可以设置卷积的步长，也就是每隔多少个元素进行一次卷积操作，我们称这个步长为 **stride**，如图3.8所示为输入大小为 7×7 的图像，**stride** 为 2 的情况，可见此时对应的输出长和宽为 $(7-3)/2+1$ 。即给定输入数据大小 (D_h, D_w, D_c) ，**kernel** 大小 (k_h, k_w) ，**stride** 大小 s ，要求卷积神经网络的隐层节点为 K ，隐层大小为 $((D_h - k_h)/s + 1, (D_w - k_w)/s + 1, K)$ ，有 $k_h \times k_w \times D_c \times K$ 个参数。

2. 降采样层

1998 年，Yan Lecun 等人提出，在网络构建时可以在 CNN 网络的卷积层后面跟一个

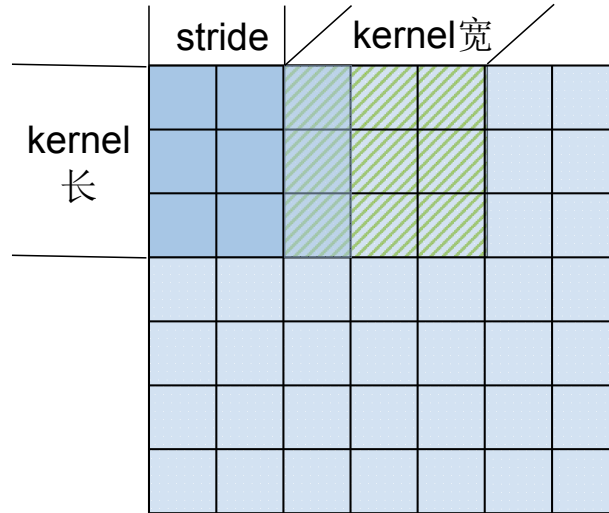


图 3.8 stride 示意图

降采样层，用来对卷积结果中的每个小窗口做平均或者最大化 pooling。有两个原因，一来这样可以减少下一层的数据数据规模，降低计算量；二来这样在一个小区域的节点中做平均或最大化选择可以增加模型的适应性，减少网络受旋转、平移等变换的影响。将卷积层和降采样层结合实际是受 Hubel 和 Wiesel 在生物实验上结果的影响^[2]。和卷积层类似，降采样层也有相应的窗口高、宽和 stride。经过降采样层，使得一块区域的数据被压缩，分在不同的“池子”里，因此降采样层所做的事情也叫做池化。除了均值和最大值，还可以取 L_1 , L_2 范数进行降采样。

3. 全连接与输出层

一个典型的卷积神经网络如图3.10所示。图为 Yan Lecun 的为 MNIST 数据集设计的卷积神经网络 LeNet5^[1]。应用在著名的手写 MNIST 数据集^[14]有 60000 个训练数据和 10000 个测试数据，样本大小为 32×32 ，均为数字 0 到 9，共 10 类。从图中可见，网络输入为数字图像，前两层为有 6 个高、宽均为 5 的 kernel（深度为 6）的卷积层和窗口高、宽均为 2 的降采样层。3,4 层类似是一个卷积加降采样层。此外，在卷积神经网络中还有两层全连接层，由全连接层连接的神经元两两连接，也就是说，和全连接层相连接的层间权重个数等于两层神经元个数的乘积，用来增强网络的表达能力。最后是输出层，对于 10 类分类问题，可以设置输出层为 10 个节点，最后采用 Softmax(如式3.13) 或 Sigmoid(如式3.16) 函数进行分类。在 LeNet 中，最后一层采用的 RBF 网络，达到测试误差达到 0.95%。训练过程可参照 LeNet5 的一个 demo¹。

¹<http://yann.lecun.com/exdb/lenet/>

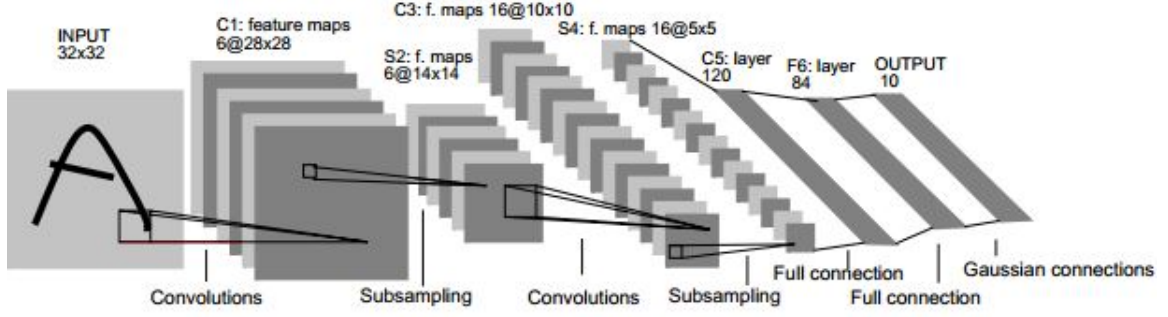


图 3.9 LeNet5, 一个为手写数字做分类的卷积神经网络。^[1]

$$S(t) = \frac{1}{1 + e^{-t}} \quad \text{公式 (3.12)}$$

$$S(t) = \frac{e^t}{\sum_{l=1}^k e^t} \quad \text{公式 (3.13)}$$

其中, t 表示输入 x 的线性组合, 即 $t = w^T x$ 。

4. 规则化层

通常, 我们希望输入数据是经过规则化 (normalization) 的, 即对数据进行白化 (whiten), 以减少数据 variance 产生的波动, 通过减均值, 除以方差完成。该层通常加在数据层之前。

5. 激励层

如果卷积神经网络中, 卷积层经过线性组合之后不加非线性函数的话, 那么整个网络就完全可以有一个线性函数来表示了。为了增强模型的表达能力, 我们加入非线性层, 使所得结果稀疏化, 或者得到侧抑制, 我们也称非线性层为激励层, 相应的非线性函数称为激励函数。通常, 我们所用的非线性函数有 rectified linear units (ReLU, 如式3.14), Component-wise shrinkage, tanh (如式3.15), sigmoid (如式3.16) 和 winner-takes-all。

$$relu(x) = \max(x, 0) \quad \text{公式 (3.14)}$$

$$\tanh(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}} \quad \text{公式 (3.15)}$$

$$\text{sigmoid}(w, x) = \frac{1}{1 + e^{w^T x}} \quad \text{公式 (3.16)}$$

综上，现在用的卷积神经网络每个卷积层经过数据规则化，卷积，非线性激励，池化（降采样）四个模块。整个卷积神经网络如图??所示。

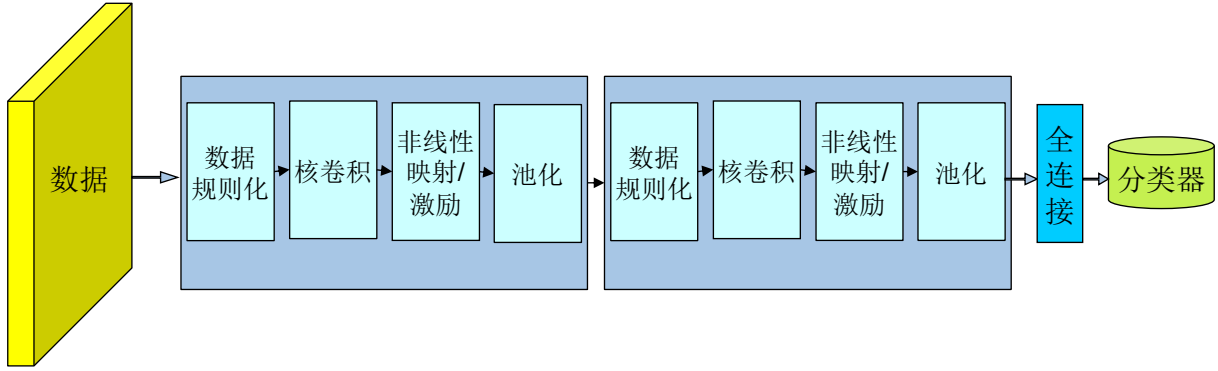


图 3.10 卷积神经网络结构示意图

3.2.2 CNN 网络训练

CNN 模型通常用随机梯度下降^[103;142] 算法进行优化。随机梯度下降算法源于随机优化。随机优化在统计学习理论中非常常见，即求参数 θ 来最小化 loss 的期望。

$$\theta^* = \operatorname{argmin}_{\theta} E(L(f(\theta, x), y)) \quad \text{公式 (3.17)}$$

在线学习中有一个简单算法求这样的 θ ，叫做在线梯度下降^[143]：在每一步迭代中，按式3.18更新权重，其中 θ_k 为第 k 步的 θ 值， η 为学习率。

$$\theta_{k+1} = \theta_k - \eta g_k \quad \text{公式 (3.18)}$$

也就是更新方向为第 k 步时梯度下降最快的方向。这种把数据看做一个源源不断的流数据，每一轮（epoch）迭代对 60000 个训练样本分别进行权重更新的方法叫做随机梯度下降（Stochastic Gradient Descent, SGD）。2006 年，Mike O'Neill 在 MNIST 上训练 CNN，采用 4 个隐层，共 3215 个隐层神经元节点，134066 个权重参数，184974 条网络连接（网络连接数大于权重参数，因为网络中有两个卷积层，其中是共享权值的）。一轮迭代大约用 40 分钟，由于收敛大约需要 30 轮迭代，所以整个训练时间大约需要 20 个小时。用该方法，他在 MNIST 上将测试集误差降到 1.4%。

3.3 ConvP300Net

在第一章中，我们已经介绍了 oddball 实验以及 P300 波形检测任务。在这一章中，我们进行第一章中提到的 P300 波形检测中的第一个任务——p300 波形二类分类，即将数据段分为是 P300 和不是 P300。之所以只做了 P300 分类而没有做字符识别是因为数据量有限，如果做 36 类分类的话数据量不够。下面我们来介绍数据集。

3.3.1 数据集

我们所用的数据来自 BCI 竞赛的第二个数据集^[144]。该数据集由纽约健康部门 Wadworth 中心提供。该数据为从两个测试个体上采集的 EEG 信号，其中有受刺激所产生的 P300 波形。生成该数据时，指定让实验主体看一个预先设定的字符，然后如图1.4 所示的矩阵以 5.7Hz 的频率进行行列交替随即闪烁。当先后交替闪烁的一行一列相交于指定字符时产生一个 P300 信号。

实验时，在最开始先空（没有任何闪烁）2.5s 做准备。然后每个行/列随即闪烁 100ms，每次闪烁后，矩阵再空 75ms。接着，每行/列再随机闪烁，对每个指定字符所在的行/列各闪烁 15 次。因此，对每个字符，理论上都有 30 个 P300 波形。信号从 64 个通道，以 240Hz 的频率进行采集，然后将信号通过 0.160Hz 的带通滤波器^[145]。对每个实验主体，训练数据和测试数据分别指定 85 个和 100 个字符，那么训练集的 P300 波形共有 85*2*15 个。根据我们上面的数据处理方法，得到训练集和测试集上的数据段个数如表3.2所示。

	训练数据	测试数据
P300	2550	3000
非 P300	12750	15000

表 3.2 各类数据数目

3.3.2 相关工作

在该数据集上，已有的一些工作主要用多分类起来提高分类精度，并用不同方法进行数据清洗。

1. 基于 SVM^[146;147]

基于 SVM 的方法^[1]

2. 基于成分分析 (ICA 和 PCA)

3. 基于 HMM^[148–150]

4. 基于 LDA

5. 基于神经网络^[151–155]

3.3.3 P300 检测网络结构

由于已有相关工作在 p300 检测上有所成效, 所以在我们的方法中, 研究采用卷积神经网络进行 P300 的检测。在数据清洗与数据预处理方面, 我们采用和 Hubert Cecotti 等人在用卷积神经网络时对数据处理相同的方法^[156]。首先对数据进行降采样, 采用 120Hz 的采样率 (而不是 240Hz) 进行数据采样。然后通过 0.1Hz 到 20Hz 的带通滤波器, 在第二章中我们已经讲过 EEG 所采集的信号主要是低频信号, 因此这样有助于减少五官噪声。过滤后的数据通过减均值除以方差来进行规则化 (normalization)。最后, 我们将数据进行分段, 并标号 (计算 label)。定义每个数据段长为 T_s , 采样率为 f , 则每个数据段持续时间 $T_s * f$ 。这里我们取 T_s 为 650ms, 即每一次从上一次闪烁和空闲之后开始采样, 采样 650ms, 共 $0.65s * 120Hz = 78$ 个采样点长的数据, 数据段之间有覆盖, 如图3.11所示, 红色的箭头表示每一个行/列闪烁的起始时刻, 黑色的箭头表示数据分段的段起始点, 每段持续 650ms, 可见数据段之间有覆盖, 每段标号以该段起始的那 100ms 闪烁为标准, 如果这 100ms 内闪烁的行/列包括指定字符, 其标号就为 1, 否则指定标号为 0。整个数据预处理过程的流程图展现在图3.12中。

我们的网络结构如图3.13所示, 我们称之为 PConv 网络。由 6 层网络 (L_0 到 L_5) 组成:

1. L_0

数据层由上面数据预处理过程得到的数据构成, 每个数据有 64 个通道, 78 个采样点组成。也就是在空间和时间维度分别为 64 维和 78 维。

2. L_1 局部连接层

L_1 局部连接层类似卷积层, 只不过卷积层 kernel 中的权重参数在空间扫描的时候是共享的, 而局部连接层是不共享的。对于大小为 (k_h, k_w) 的 kernel, 一共可以卷

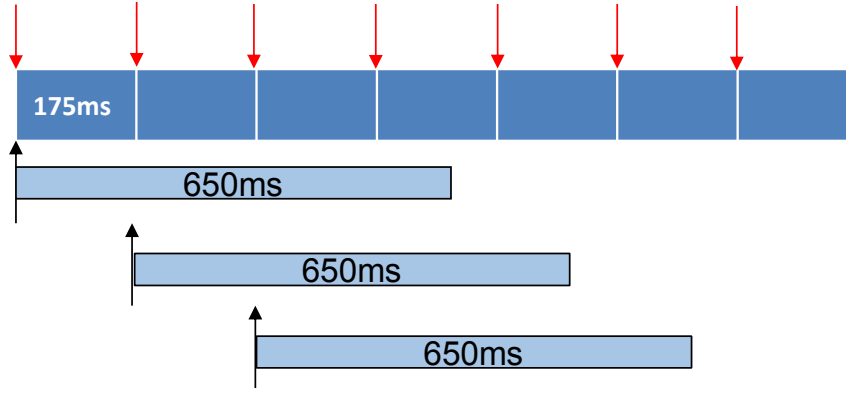


图 3.11 P300 检测——数据段分割

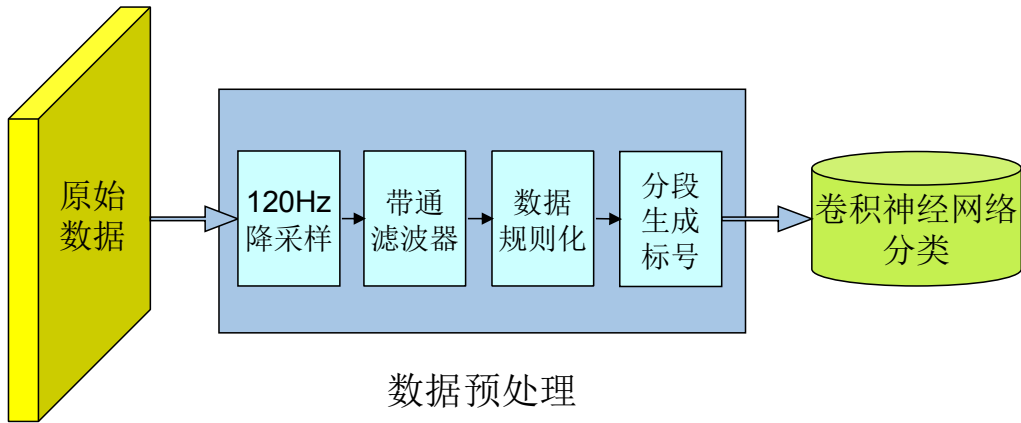


图 3.12 P300 检测——数据预处理

积 $(D_h - k_h + 1) * (D_w - k_w + 1)$ 次，对于有 K 个节点的全连接层，其参数数目为 $k_h * k_w * K$ ；对于局部连接层，其参数数目为 $(D_h - k_h + 1) * (D_w - k_w + 1) * k_h * k_w * K$ 个。在 Cecotti 的工作中^[156]，第一层数据输入后采用卷积层将样本进行操作，卷积核大小为 $64 * 1$ ，即在空间域对数据进行卷积，在时间维度共享这个卷积核的权重。但是，时间维度为我们人为划分的数据段，并没有道理说，所有时间点上 64 个通道起相同作用，即不应该让不同时间采样点共享权重。所以在我们的工作中，依然采用 $64 * 1$ 的核函数，但是在不同时间点有不同的权重大小，即第一层采用局部连接层对空间域进行卷积。

3. L_2 卷积层

在 L_2 ，我们采用卷积层将 L_1 的输出进行卷积。对于大小为 $N_s * 1 * 78$ 的输入，卷积核大小为 $1 * 1$ ，也就是在空间域分别对每一个神经元进行卷积，设置 $5N_s$ 个卷积核，其中 N_s 为 L_1 的节点数（深度）。然后，我们在 L_2 后面接一层降采样层和激励层（考虑到画图简洁性并未在图中画出）。降采样层分别在高、宽进行长为 13, 1 的降采样，

降采样 stride 为 13，也就是进行无重合的降采样，并将得到的结果输入激励层进行非线性映射，这里我们采用非线性函数 $Relu$ 。

4. L_3 全连接层

全连接层设置 100 个节点，与 L_2 全连接。经过降采样， L_2 输出共有 $5N_s * 6$ 个节点，设置全连接层有 100 个节点，那么 L_3 中共有 $5N_s * 6 * 100$ 个连接。

5. L_4 全连接层

和 L_3 层类似， L_4 也设置为全连接层，由于是二类分类，我们设置 L_4 为预测输出，即只有两个节点，一个表示正样本（有 P300 波形），一个表示负样本（没有 P300 波形）。在预测时，我们取前向网络计算所得值大的作为类别输出。在训练时，我们将其输入 L_5 loss 层。

6. L_5 Loss 层

在训练时，最后我们将预测样本类别概率与真实样本标号输入 loss 层求 loss 以便通过误差反传算法训练网络。

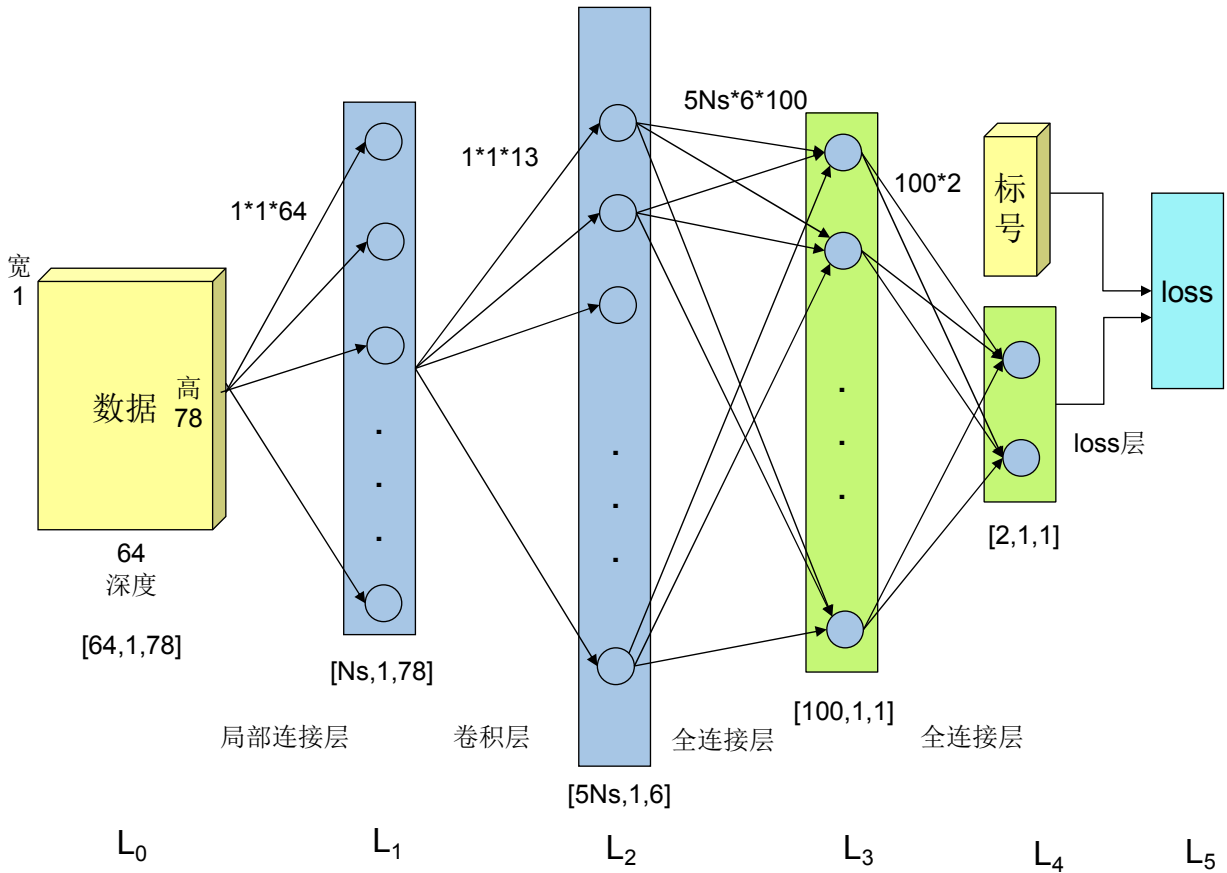


图 3.13 ConvP300Net

3.4 实验结果及实现

3.4.1 PConv 的学习

整个网络的学习采用误差反传算法，从后往前对每一层分别求导将误差传回，如第3.1.3.2节所示。但有两点不同，一是在 PConv 中，我们采用不同的 loss 函数，二是选用 Relu 非线性映射，但是该函数在零点不可导，下面我们来分别说明。

1. Loss 层的设计

由表3.2可知，该样本存在正负样本不均衡的问题，正样本与负样本比例为 1: 5。因此我们考虑样本均衡方法，如 Hubert 的工作中分别设置了 5 个分类器，每个分类器用全部正样本和 $\frac{1}{5}$ 的负样本做训练。但是，这样还需要考虑到后续分类器做融合时 ensemble learning 带来的提升效果。为了在同一个网络中进行训练并作性能测试，我们更改 loss 层定义，改用 weighted loss。在 weighted loss 中，我们受 boosting 启发^[1]，赋给正样本较大权重，赋给负样本较小权重。这样在调整的时候就可以更注重正样本的调整。采用 euclidean loss, 其 loss 定义如下：

$$L(f(x), y) = \sum_{i=1}^N g(y_i)(f(x_i) - y_i)^2 \quad \text{公式 (3.19)}$$

其中 $g(y)$ 为根据 y 设置的权重：

$$g(y) = \begin{cases} \frac{1}{6}, & \text{if } y = 1 \\ \frac{5}{6}, & \text{if } y = -1 \end{cases}$$

所以，

$$L(net_n, y) = \frac{1}{2} \left[\frac{5}{6}(y + 1) + \frac{1}{6}(1 - y) \right] (net_n - y)^2 \quad \text{公式 (3.20)}$$

而后其更新可以根据算法3得到, 第 n 层的反向传播误差为：

$$\delta_n = \frac{\partial L}{\partial net_n} = \left[\frac{5}{6}(y + 1) + \frac{1}{6}(1 - y) \right] (net_n - y) \quad \text{公式 (3.21)}$$

而对之前的隐层节点 net_j , 通过链式求导法则可得，

$$\delta_j = \sum_k \delta_k \cdot w_{jk} f'(net_j) \quad \text{公式 (3.22)}$$

其中 k 是所有与 j 节点相连的下一层节点编号。

2. 非线性映射 Relu

神经网络的非线性激活函数有多种选择，sigmoid, tanh 和 ReLU 的表达式我们已经在上文中介绍过，其图形如图3.14所示。很多工作中显示 ReLU 有助于提升卷积神经网络的效果^[2,7]。其区别主要是 sigmoid 函数的值域在 $[0,1]$ 而 ReLU 的值域在 0 到正无穷。所以 sigmoid 可以用来为概率建模，而 ReLU 可以为正实数建模。然而，从图3.14可知 ReLU 在 0 处不可导，所以 ReLU 采用分段求导 ($x < 0$ 时 $g'(x)=0$, $x > 0$ 是 $g'(x)=1$)。

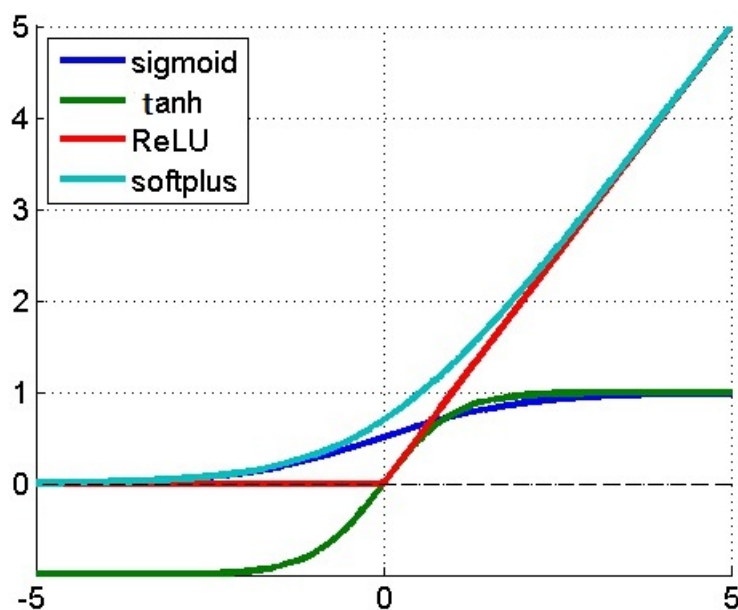


图 3.14 激励函数 sigmoid,tanh,ReLU,softplus

相比其他非线性映射函数，ReLU 有一些显著优势。首先，它具有 sparse 约束性，因为从概率角度，只有 50% 的神经元被激活。其次，ReLU 具有没有 tanh, sigmoid 的梯度 vanishing 和 exploding 问题^[2]。最后，ReLU 具有计算简单的性质，所以更为高效。

3.4.2 实验结果

在实验中，我们将我们的方法与 Hubert Cecotti 的结果做以比较，用了以下评价标准：

1. TP, true positive 的缩写，表示分正确的正样本数。
2. TN, true negative 的缩写，表示分正确的负样本数。
3. FP, false positive 的缩写，表示错分成正样本（实际是负样本）的数目。

4. FN, false negative 的缩写, 表示错分成负样本 (实际是正样本) 的数目。

5. Reco., recognition rate 表示识别率, 表示所有样本中分对的比例。

6. Recall, 召回率表示分对的正样本数占有所有分对样本数的比例, 即

$$Recall = \frac{TP}{TP + TN} \quad \text{公式 (3.23)}$$

7. Precision, 精度表示分对的正样本数占有所有估计成正样本的样本中的比例, 即

$$Precision = \frac{TP}{TP + FP} \quad \text{公式 (3.24)}$$

8. F-value, 作为一个综合评价指标, F-value 将 Recall 和 Precision 都考虑了进来:

$$F - value = 2 \frac{Recall \cdot Precision}{Recall + Precision} \quad \text{公式 (3.25)}$$

采用各网络时, A, B 两个人的结果分别如表3.3, 3.4所示。其中, CNN1, CNN3, MCNN1, MCNN3^[156] 的基本模型一样, 且也是采用卷积神经网络对数据进行建模。不同之处是, L_1 采用卷积神经网络, 在数据 78 个时间点的每个时间点共享空间域 64 个通道的权值; L_2 采用先降采样后卷积 (而非我们方法中的先卷积后降采样) 的策略以降低模型复杂度, 且采用 tanh 作为激励函数 (而我们的 PCNN, PsCNN, PwCNN, PConv 方法均选用 Relu)。我们将该模型记为 CNN1。在 CNN1 的基础上, CNN3 在 L_1 中只设置一个节点, 也就是说 L_1 的深度为 1。此外, 如表3.2所示, 该样本存在正负样本不均衡的问题, 正样本与负样本比例为 1: 5。为解决该问题, MCNN 中提出用多个分类器。在 MCNN1 中, 将所有负样本分为 5 份, 用 5 个分类器分别训练全部正样本加 $\frac{1}{5}$ 的负样本。在 MCNN3 中, 为了避免网络随机参数初始化带来的波动问题选用了 3 个分类器中最好的结果作为最后的分类结果。与 CNN1 相比, 我们的 PCNN 方法将 L_2 中的降采样与卷积交换顺序, 即先卷积, 后降采样, 这是因为参考其他网络 (如 LeNet, AlexNet) 都是先做卷积再做 pooling, 而且这样可以提高模型参数规模, 更好地拟合数据; PsCNN 在 PCNN 的基础上在 L_4 输出概率后加了一层非线性映射, 希望提高模型适应性; PwCNN 是 PCNN 加了 weighted loss 版本, 详见第3.4.1节; 在 PwCNN 的基础上, PConv 模型将第一层从卷积层改为局部连接层, 使得 64 维通道的权重在不同时间点不共享。

我们将 PConv, PwCNN, PsCNN 的训练 loss 随迭代次数下降的曲线画出来, 如图3.15所示。可见, PConv 和 PwCNN 的训练 loss 下降曲线类似, 而加了 sigmoid 非线性映射后, PsCNN 很快下降后几乎损失函数不降了。这是因为加了 sigmoid 后原本概率被映射到 [0,1], 使得误差非零即一。大约 5 次迭代后达到收敛。

	TP	TN	FP	FN	Reco.(%)	Recall	Precision	F-value
CNN1	2021	10645	4355	979	70.367	0.674	0.317	0.431
CNN3	1827	10554	4446	1173	68.783	0.609	0.291	0.394
MCNN1	2071	10348	4652	929	68.994	0.690	0.308	0.426
MCNN3	2023	10645	4355	977	70.378	0.674	0.317	0.431
PCNN	2079	10597	4403	921	70.422	0.693	0.321	0.439
PsCNN	2013	10483	4517	987	69.422	0.671	0.308	0.422
PwCNN	2194	11071	3929	806	73.694	0.731	0.358	0.481
PConv	2216	11402	3598	784	75.656	0.739	0.381	0.503

表 3.3 实验人 A ——P300 检测各卷积神经网络结果

	TP	TN	FP	FN	Reco.(%)	Recall	Precision	F-value
CNN1	2035	12039	2961	965	78.189	0.678	0.407	0.509
CNN3	2006	11348	3652	994	74.189	0.669	0.355	0.463
MCNN1	2202	11453	3547	798	75.861	0.734	0.383	0.503
MCNN3	2077	11997	3003	923	78.189	0.692	0.409	0.514
PCNN	2145	12114	2886	855	79.217	0.715	0.426	0.534
PsCNN	2194	12383	2617	806	80.983	0.731	0.456	0.562
PwCNN	2276	12745	2255	724	83.450	0.759	0.502	0.604
PConv	2537	12679	2321	463	84.533	0.846	0.522	0.646

表 3.4 实验人 B ——P300 检测各卷积神经网络结果

将 PConv, PwCNN, PsCNN 的测试 loss 随迭代次数变化的曲线画在图3.16。如图可见，在测试集上的 loss 总体上呈下降趋势，但可能存在局部波动。最后，我们将三种方法的测试准确率展现在图3.17中。可见，三种方法的测试准确率都较为稳定，PConv 的准确率最佳。

3.4.3 实现方法

我们的实现基于 Berkeley Vision and Learning Center 的 Yangqing Jia 开发实现的 Caffe^[157]，为开源代码。它可以实现在 CPU 和 GPU 上运行，这里我们选用的 GPU 为 NVIDIA 出的 GT Force 580，基于 cuda 进行并行加速。Caffe 用纯 C++ 实现，提供了 Python，

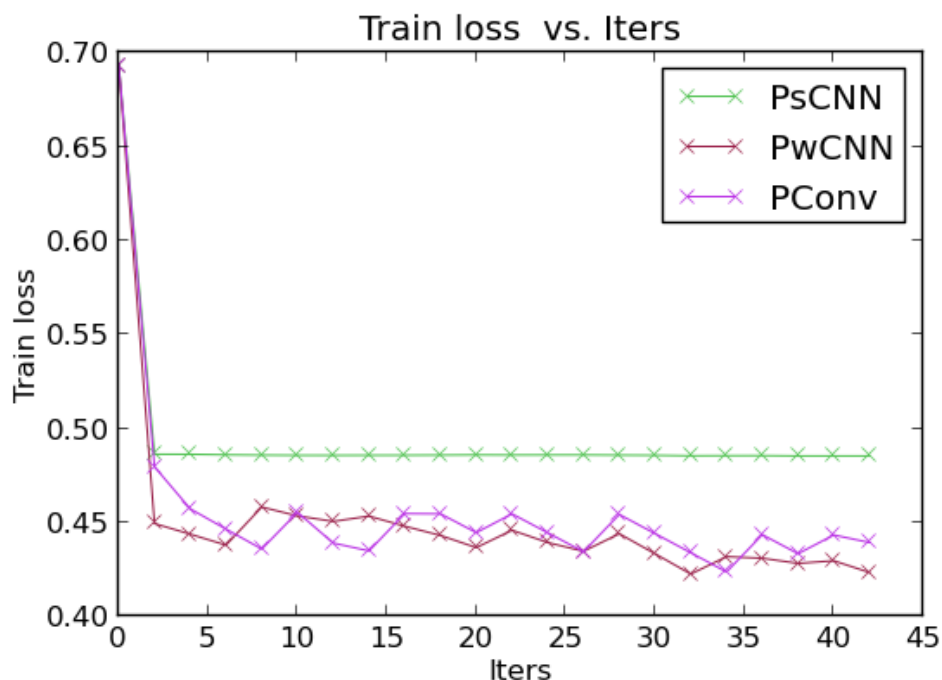


图 3.15 ConvP300Net 训练 loss-迭代次数

Matlab 脚本和命令行接口。通过配置文件可以定义网络结构和参数配置。实验中，我们将数据经过以下处理，先将所有数据分为多个 batches。如 15300 个训练数据分为 100 个 batch，其中 95% 作为训练数据，5% 作为测试数据，每个 batch 有 153 个数据。然后经过 leveldb 将数据转换成 kv 存储，然后在网络中进行训练。数据在整个网络中都通过块 (blob) 的形式定义。在 caffe 的基础上，本文在数据层进行数据生成与转换，在 layer 中增加了 weighted euclidean loss 层，并调整网络结构。

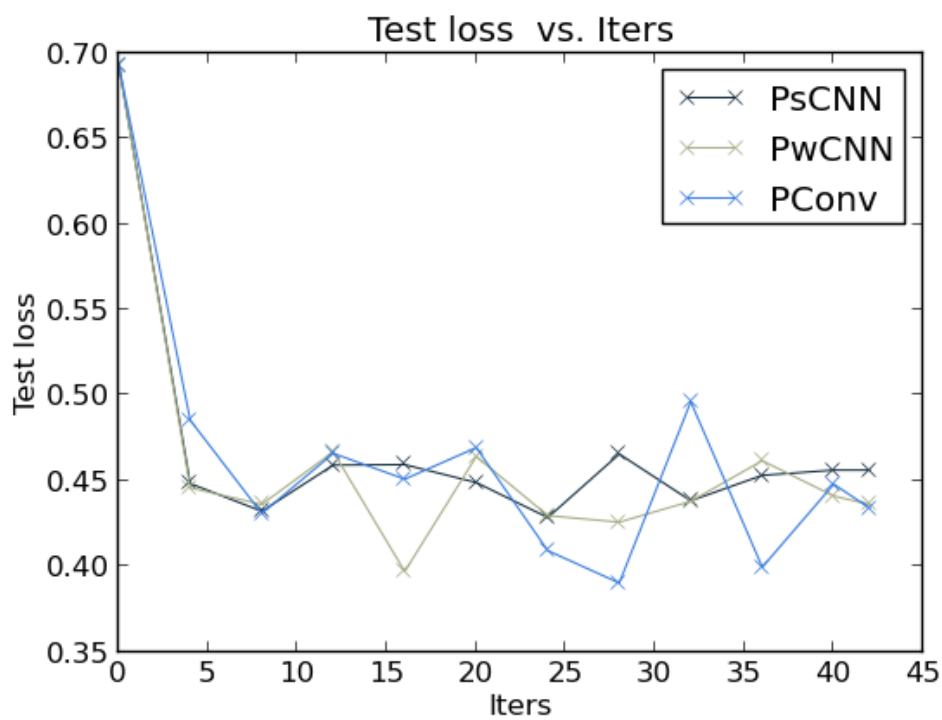


图 3.16 ConvP300Net 测试 loss-迭代次数

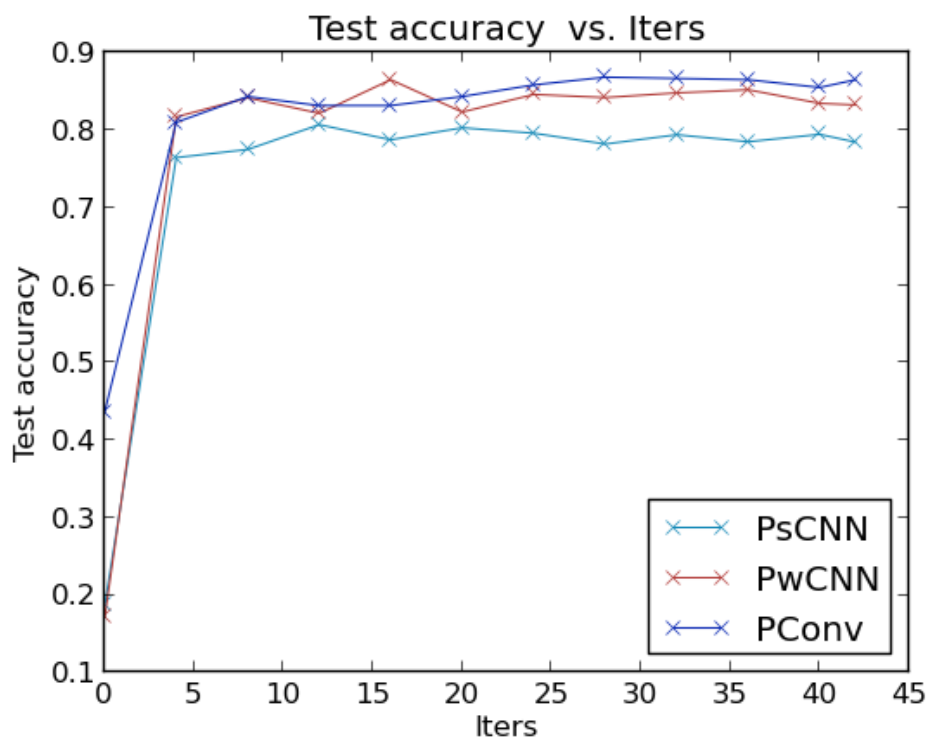


图 3.17 ConvP300Net 测试准确率-迭代次数

第4章 基于循环神经网络的信号解码

4.1 循环神经网络 (RNN)

人类可以很快地识别节奏模式序列，即那些有时间间隔的子模式。此外，鼓手等演奏者也能根据运动命令生成有精确时间节奏的节奏序列。这促使人们研究人工系统，去分离或产生通过事件间时间间隔长度传递信息的模式。

Deep Neural Network(DNN) 通常在较难的 learning 问题上能够达到比较好的效果。在实际问题中有很多是时序信号，如语音识别, 连续（即字符无分割的）手写识别, 蛋白质分析, 股市预测等。虽然这些问题无论是否有监督，都可以交给 CNN 或者 DNN 处理并达到较好效果 [83;158? ? ? ;159]，但是并不能完全利用上信号的时序性，而循环神经网络（recurrent neural network）可以利用神经元循环传递实现时序信号的分析。

一个循环神经网络是一个结构中带反馈连接的神经网络，它可以学习并处理时序序列问题，而传统的机器学习算法由于没有考虑时间方面相关性可能并不善于解决这类问题。之前对于这种时序信号的处理方法是一些基于可学习可适应的方法，如隐马尔可夫模型 [160], 前向网络等，但是在计算能力和物理意义上 RNN 都比这些方法能力强。而且事件在时域上的信息很大程度上反映了时序任务的重要信息，如运动控制和节奏检测。隐马尔可夫模型往往忽视这一信息，而循环神经网络（Recurrent Neural Network, RNNs）可以从根本上学习应用这个信息。隐马尔可夫模型（HMM）能够成功地应用在语音识别，正是因为这种方法不喜欢特定口语单词语速产生的差异。而如节奏检测，音乐处理，和其他本文中所述的任務，都需要精确的时间测量。虽然 HMM 可以通过给每个时间间隔单独创建一个内部状态来解决有限个时间间隔的问题，但是这样既麻烦又低效的，并且没有用到 HMM 的非线性时序伸展不变性。

事实上，尽管隐马尔可夫模型和传统的离散符号语法学习设备被限制在离散状态空间，RNNs 在根本上上适用于所有的序列学习任务，因为它们具有图灵能力 [161]。典型的 RNN 学习算法在一个潜在抗噪算法的通用空间中进行梯度下降 [162]，这个通用算法采用分布式，用连续值表示的内部状态来将实值的输入序列映射为实值输出序列。所以循环神经网络在识别由时间距离定义的模式上更有希望。混合 HMM 的 RNN 方法 [163] 可能能够结合两种方法的优点，但就我们所知，从未被用到准确的事件计时的問題。

然而，Schmidhuber 的学生 Hochreiter 在他的博士论文 [164] 中指出了 RNN 的缺点：

vanishing or exploding gradients: 在传统 RNN 中, 反传的误差要不就收缩太快, 要不就爆发超出界限。事实上误差会随着网络层数而指数型衰减或爆发, 因此很难训练, 而且不能处理长时间延时效件。

之后 Hochreiter 提出了一种新型的 RNN——长短期记忆 Long Short-Term Memory^[165], 它在涉及很长一段时间延时的任务上比传统 RNNs 更好。LSTM 网络中的基本单元是包含一个或多个存储单元 (memory cell) 和所有 cell 共享的三个自适应乘法门 (图 2) 的存储块 (memory block)。每个存储单元都在其中心有一个自连接单元, 我们称之为“恒误差旋转木马” (Constant Error Carousel, CEC), 它本质上是一个无参数的线性单元。通过无限循环地进行激活和误差信号传递, CEC 可以在延长的时间段内提供短期存储。Gers et al 在 LSTM 中安置了忘记门^[166]。输入, 忘记和输出门可以通过训练分别进行学习, 即学会将什么信息存入, 怎样保存, 什么时候读出。

LSTM 的体系结构允许 LSTM 容忍输入的事件之间有很长的时间延时 (1000 步及以上); 而传统 RNN 用代价更高的更新算法, 比如 BPTT^[167], RTRL^[168;169], 或它们的组合^[170;171], 都无法学习超过 10 个时间延时的序列^[164-166;172;173]。

比如, 一些我们以前任务需要用 LSTM 网络对发生于 50 个离散时间间隔之前的事件予以响应, 而不受发生于之前 49 个时间间隔的事件的影响。就在关键时刻之前, 如果有一个有用的“标记”输入通知网络, 其下一步的行动将是至关重要的。因此, 网络没有必要学习度量 50 个步骤的时间间隔; 它只需要学会保存这 50 步的相关信息, 而后一旦观察到这个标记就用这段信息即可。

但是如果没有这样的标记呢? 如果网络本身需要学习测量并在内部表示这段特定任务的间隔内信息, 或生成由特定时间间隔分割的序列模式, 就要求在长时间连续输入流中保持网络的时间精确性和鲁棒性。显然, 这样的任务通常不能由基于共同时间窗口的方法解决的, 因为它们的泛化能力受时间窗口大小的限制。不幸的是, 这意味着我们不能用标准基准进行测试和计时, 因为就我们所知, 目前没有任何其他网络学会将 15 个时间滞后推广到 45 个等。因此, 我们不得不以创建一组新的比较任务。那将用到 LSTM 了。

因为长短期记忆方法已被证明在长时间滞后任务上比其他 RNN 的方法强, 而且其体系结构与人脑对信息的处理方式很类似, 考虑到这种方式可能可以更好地模拟大脑皮层信号激发方式, 所以我们研究该方法。LSTM 通过内部细胞到多门的“窥孔连接”, 在没有任何段式训练样本的情况下可以很好的区分 50 个和 49 个时间间隔的 spike 序列。在没有外部复位和监督学习的情况下, LSTM 的变种也能学习生成时间精确的稳定 spike 流和其他非线性周期模式。这使得 LSTM 在需要的精确测量或生成有时间间隔的任务上很有效。

4.2 LSTM 的应用

我们先来看一下 LSTM 的已有应用：

1. 语音识别 LSTM RNN 是著名语音识别数据库 TIMIT 的 benchmark (Graves et al, ICASSP 2013), Nicole 等人用 LSTM 网络进行 RNN 的 retraining^[174], 这是无法在 HMM 中实现的。google 用 LSTM RNN 来改善大规模语音识别^[175]。在类似问题上,^[176] 将 LSTM 应用在机器翻译问题, 实现了 sequences to sequence 学习。
2. 连续 (即字符无分割的) 手写识别 Connectionist Temporal Classification (CTC)^[177] 训练的 RNN (CTC-LSTM)^[178], 在 2009 年跑得结果赢了很多手写识别竞赛的冠军^[179]。
3. 音乐合成 通过训练, RNN 的变种 RNN-RBM 可以无监督地形成简单的钢琴旋律。[5] Modeling Temporal Dependencies in High-Dimensional Sequences: Application to Polyphonic Music Generation and Transcription. 通过用音乐集训练 LSTM 网络, 网络就可以无监督地生成有特定样式的旋律与和弦的音乐样本。而前向网络 and 传统 RNN 都无法学习出和弦音。^[180]
4. 蛋白质分析 在 RNN 中还可以实现无分割的蛋白质分析^[181], 其本质和无分割的手写体识别同理。

4.3 循环神经网络网络结构

4.3.1 RNN 网络结构

首先我们来看 RNN 的结构: RNN 包含一些用带权连接的单元, 每个单元有一个在时间 t 更新的激活函数 $y(t), t = 1, 2, \dots$ 。对单元 i 的激活 y^i 通过计算网络中与 i 的相连输入 $net^l(t)$ 得到:

$$net^l(t) = \sum_m w_{im} y^m(t-1) \quad \text{公式 (4.1)}$$

而后通过一个可导函数 f 将 $net^l(t)$ 进行”压平” (f 比如我们之前提到的 sigmoid, tanh, 或 Relu 函数):

$$y^i(t) = f(net^i(t)) \quad \text{公式 (4.2)}$$

其中网络的输入是随时间变化的序列, 对于传统的 RNN 回归问题或者序列分类问题 (如手写体识别), 输出也是随时间变化的序列。而对于纯分类问题, 输出只是一个标量。为了方便讲述传统 RNN 的方法, 我们在这里使用输入输出都是时间序列来描述。

对有监督的 RNN 而言，当采用平方和误差作为 loss 时，我们将每个单元的损失计为与其输入与真实输入之差，而全局损失函数为所有样本损失的平方和，即 t 时刻的总误差为：

$$\begin{aligned} E(t) &= \frac{1}{2} \sum_i e_i(t)^2 \\ e_i(t) &:= Y^i(t) - y^k(t) \end{aligned} \quad \text{公式 (4.3)}$$

整个序列的误差为每一时刻的误差 $E(t)$ 之和。

和 CNN 类似，RNN 可以用梯度下降进行网络参数更新，即对于每个权重 w_{im} ，有 $\partial w_{im} = -\alpha \frac{\partial E(t)}{\partial w_{im}}$ ，其中 α 为学习率。这种方法也是标准 RNN 的更新方法 backpropagation through time (BPTT) 方法，通过带入可得 t 时刻 i 单元的误差信息为

$$\partial e_i(t) = f'_i(\text{net}_i(t)) \sum_j w_{ji} e_j(t+1) \quad \text{公式 (4.4)}$$

然而在 1990 年，Schmidhuber 的学生 Hochreiter 提出 RNN 在理论上很好，但是不好训练，误差会以指数增长或者消失^[171]，原因是 BPTT，real-time recurrent learning (RTRL)^[169]，这类基于梯度下降的方法都有一个问题，如果计算权重更新后的 q 步，则如公式 4.5 所示， t 时刻与 $t-q$ 的更新误差之比与 w_{lv} 有关，

$$\frac{\partial e_v(t-q)}{\partial e_u(t)} = \begin{cases} f'_v(\text{net}_v(t-1))w_{uv}, q=1 \\ f'_v(\text{net}_v(t-q)) \sum_{l=1}^n \frac{\partial e_l(t-q+l)}{\partial e_u(t)} w_{lv}, q>1 \end{cases} \quad \text{公式 (4.5)}$$

将上式 $q>1$ 的情况展开，用 l_t 表示 t 时刻的该节点， $l_q = v, l_0 = u$ ，可得：

$$\frac{\partial e_v(t-q)}{\partial e_u(t)} = \sum_{l_1=1}^n \dots \sum_{m=1}^q \prod_{m=1}^q f'_{l_m}(\text{net}_{l_m}(t-m)) w_{l_m l_{m-1}} \quad \text{公式 (4.6)}$$

其中 $\prod_{m=1}^q f'_{l_m}(\text{net}_{l_m}(t-m)) w_{l_m l_{m-1}}$ 决定了总的传回误差。所以如果 $|f'_{l_m}(\text{net}_{l_m}(t-m)) w_{l_m l_{m-1}}| > 1.0$ ，则最大的乘积随 q 指数增长，就会使学习过程不稳定；相反地，如果 $|f'_{l_m}(\text{net}_{l_m}(t-m)) w_{l_m l_{m-1}}| < 1.0$ ，则最大的乘积随 q 指数下降以至消失，在有限时间内无法学到有效网络参数，这也被称为长时间延时问题 (long time lag problem)，使得 RNN 的应用一度局限在短时间序列，而我们在绪论中提到的长短时间记忆方法有效克服了这一问题，我们将在下一节中予以介绍，关于 RNN，更多优化方法请参考 Bengio 在^[182]中的综述。

4.3.2 LSTM 的网络结构

LSTM 通过设置恒定的传递误差来解决上面提到的问题。在基本 LSTM 单元中，LSTM 引入了一个自连接线性单元，称作误差旋转木马 (error carousel)，恒定的传递误差中设定

权重 w 恒等于 1, f 为线性函数, 目的是这各部分只起到传递误差的作用, 而将学习参数作用于其他权重。

粗略地表示, 我们在第一章中简单介绍了 LSTM。它以 memory block 作为基本单元, 其中包含一个至多个 memory cell, 典型的 LSTM memory cell 如图 4.1 所示: 核心有一个线性单元 (或者称作线性神经元, 如图橘黄色所示), 这是个线性单元, 在任意给定时间, 该单元将其所有输入通过加权连接求和。它的自循环连接权重固定为 1 (即与左边紫色圆点之间的半圆连接权重固定为 1), 这样通过确保训练信号或者误差在被传递的时候不会消失 (vanish) 克服了原先 RNN 的一个主要问题: vanish (参考 3.2 节)。而且这个核心神经元为线性让 LSTM 实现了可以识别多达 1000 步之前的模式。

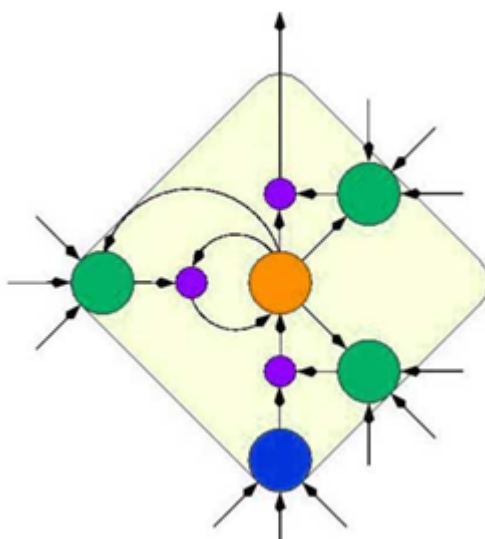


图 4.1 LSTM 模型

在这个核心周围是非线性可适应神经元, 用来学习非线性神经动作。图中蓝色神经元用来表示输入, 三个绿色神经元分别表示三个门 (左边的是忘记门, 右下角为输入门, 右上角为输出门), 这些门通过学习来使核心线性神经元不受无关事件和误差信号的干扰, 而又能通过有效信息更新网络参数, 即门是否打开来决定信号的有效性。最后, 紫色圆点用以表示乘积操作。LSTM 的学习算法是非常高效的, 每个边权的计算复杂度不超过 $O(1)$ 。

LSTM 的应用中, 可以将多个 memory cell 进行组合, 如图 4.2 所示混合模型。其中有两个 memory cell, 分别从输入和输入门接受输入, 每个 cell 将输出结果分别传给另一个 memory cell 的输入, 输入门, 输出门和忘记门。但实际应用中, 通常将一至多个 memory cell 组合成 memory block, 共享输入, 输出, 忘记门的权重 (而不共享输入的权重), 这样可以减少自适应参数的数量, 同时使 block 内部各 cell 起到不同作用。

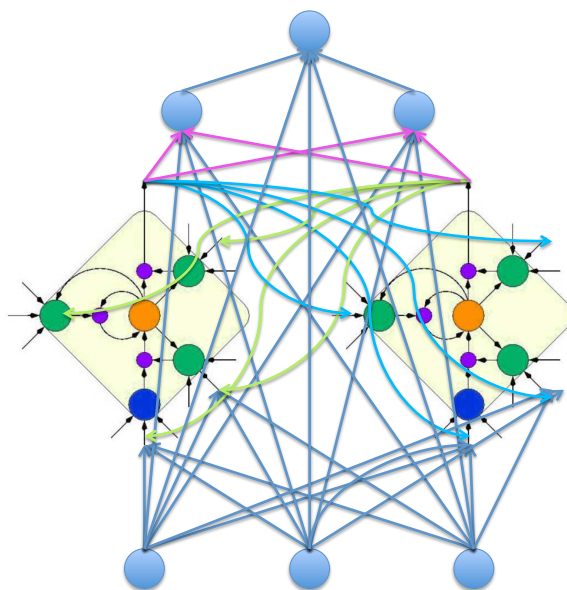


图 4.2 LSTM 混合模型

4.4 LSTMP300Net

在这一节中，我们利用 LSTM 进行 P300 分类。如第三章数据集介绍，这里我们应用同样的 P300 波形进行 P300 检测。由于 P300 信号为采集的时序信号，所以采用时序模型更为合适。

第 5 章 总结

本文

参考文献

- [1] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11):2278–2324.
- [2] Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, Theresa M Vaughan. Brain–computer interfaces for communication and control[J]. Clinical neurophysiology, 2002, 113(6):767–791.
- [3] Marcel van Gerven, Jason Farquhar, Rebecca Schaefer, Rutger Vlek, Jeroen Geuze, Anton Nijholt, Nick Ramsey, Pim Haselager, Louis Vuurpijl, Stan Gielen, et al. The brain–computer interface cycle[J]. Journal of Neural Engineering, 2009, 6(4):041001.
- [4] John P Donoghue. Connecting cortex to machines: recent advances in brain interfaces[J]. nature neuroscience, 2002, 5:1085–1088.
- [5] Aleksander Kostov, Mark Polak. Parallel man-machine training in development of eeg-based cursor control[J]. Rehabilitation Engineering, IEEE Transactions on, 2000, 8(2):203–205.
- [6] Brendan Z Allison, Elizabeth Winter Wolpaw, Jonathan R Wolpaw. Brain-computer interface systems: progress and prospects[J]. 2007.
- [7] Niels Birbaumer, Leonardo G Cohen. Brain–computer interfaces: communication and restoration of movement in paralysis[J]. The Journal of physiology, 2007, 579(3):621–636.
- [8] Benjamin Blankertz, Guido Dornhege, Matthias Krauledat, K Muller, Volker Kunzmann, Florian Losch, Gabriel Curio. The berlin brain-computer interface: Eeg-based communication without subject training[J]. Neural Systems and Rehabilitation Engineering, IEEE Transactions on, 2006, 14(2):147–152.
- [9] Fabien Lotte, Marco Congedo, Anatole Lécuyer, Fabrice Lamarche. A review of classification algorithms for eeg-based brain–computer interfaces[J]. Journal of neural engineering, 2007, 4.
- [10] Klaus-Robert Müller, Michael Tangermann, Guido Dornhege, Matthias Krauledat, Gabriel Curio, Benjamin Blankertz. Machine learning for real-time single-trial eeg-analysis: from brain–computer interfacing to mental state monitoring[J]. Journal of neuroscience methods, 2008, 167(1):82–90.
- [11] Wei Wu, Michael J Black, Yun Gao, Elie Bienenstock, Mijail Serruya, Ali Shaikhouni, John P Donoghue. Neural decoding of cursor motion using a kalman filter[J]. Advances in neural information processing systems, 2003, 133–140.
- [12] M Yu Byron, Caleb Kemere, Gopal Santhanam, Afsheen Afshar, Stephen I Ryu, Teresa H Meng, Maneesh Sahani, Krishna V Shenoy. Mixture of trajectory models for neural decoding of goal-directed movements[J]. Journal of neurophysiology, 2007, 97(5):3763–3780.

- [13] Caleb Kemere, Krishna V Shenoy, Teresa H Meng. Model-based neural decoding of reaching movements: a maximum likelihood approach[J]. Biomedical Engineering, IEEE Transactions on, 2004, 51(6):925–932.
- [14] Tomoyasu Horikawa, Masako Tamaki, Yoichi Miyawaki, Yukiyasu Kamitani. Neural decoding of visual imagery during sleep[J]. Science, 2013, 340(6132):639–642.
- [15] Simon J Thorpe, Michèle Fabre-Thorpe. Seeking categories in the brain[J]. SCIENCE-NEW YORK THEN WASHINGTON-, 2001, 260–262.
- [16] Lawrence Ashley Farwell, Emanuel Donchin. Talking off the top of your head: toward a mental prosthesis utilizing event-related brain potentials[J]. Electroencephalography and clinical Neurophysiology, 1988, 70(6):510–523.
- [17] Anne Hsu, Alexander Borst, Frédéric E Theunissen. Quantifying variability in neural responses and its application for the validation of model predictions[J]. Network: Computation in Neural Systems, 2004, 15(2):91–109.
- [18] Alexander Kraskov, Rodrigo Quiñero, Leila Reddy, Itzhak Fried, Christof Koch. Local field potentials and spikes in the human medial temporal lobe are selective to image category[J]. Journal of cognitive neuroscience, 2007, 19(3):479–492.
- [19] Bruno B Averbeck, Peter E Latham, Alexandre Pouget. Neural correlations, population coding and computation[J]. Nature Reviews Neuroscience, 2006, 7(5):358–366.
- [20] Jason S Prentice, Jan Homann, Kristina D Simmons, Gašper Tkačik, Vijay Balasubramanian, Philip C Nelson. Fast, scalable, bayesian spike identification for multi-electrode arrays[J]. PloS one, 2011, 6(7):e19884.
- [21] David A Huffman, et al. A method for the construction of minimum redundancy codes[J]. Proceedings of the IRE, 1952, 40(9):1098–1101.
- [22] Warren Hunt, Bertrand A Maher, Doug Burger, Kathryn S McKinley. Optimal huffman tree-height reduction for instruction-level parallelism[J]. 2008.
- [23] QiaoSheng Zhang, ShaoMin Zhang, YaoYao Hao, HuaiJian Zhang, JunMing Zhu, Ting Zhao, JianMin Zhang, YiWen Wang, XiaoXiang Zheng, WeiDong Chen. Development of an invasive brain-machine interface with a monkey model[J]. Chinese Science Bulletin, 2012, 57(16):2036–2045.
- [24] Monica Fira, Liviu Goras. Biomedical signal compression based on basis pursuit[C]. In Proceedings of the 2009 International Conference on Hybrid Information Technology. ACM, 2009, 541–545.
- [25] Rodrigo Quiñero, Zoltan Nadasdy, Yoram Ben-Shaul. Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering[J]. Neural computation, 2004, 16(8):1661–1687.
- [26] Josh Coalson. Flac-free lossless audio codec[J]. Internet: <http://flac.sourceforge.net>, 2008.
- [27] Chen Han Chung, Yu-Chieh Kao, Liang-Gee Chen, Fu-Shan Jaw. Intelligent content-aware model-free low power evoked neural signal compression. In Advances in Multimedia Information Processing-PCM 2008, Springer, 2008, 898–901.

- [28] Chen Han Chung, Liang-Gee Chen, Yu-Chieh Kao, Fu-Shan Jaw. Multichannel evoked neural signal compression using advanced video compression algorithm[C]. In Neural Engineering, 2009. NER'09. 4th International IEEE/EMBS Conference on. IEEE, 2009, 697–701.
- [29] Selin Aviyente. Compressed sensing framework for eeg compression[C]. In Statistical Signal Processing, 2007. SSP'07. IEEE/SP 14th Workshop on. IEEE, 2007, 181–184.
- [30] Frank Rosenblatt. Perceptron simulation experiments[J]. Proceedings of the IRE, 1960, 48(3):301–309.
- [31] Sangoh Jeong, Chee Sun Won, Robert M Gray. Image retrieval using color histograms generated by gauss mixture vector quantization[J]. Computer Vision and Image Understanding, 2004, 94(1):44–66.
- [32] Robert M Gray. Gauss mixture vector quantization[C]. In Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on. IEEE, 2001, volume 3, 1769–1772.
- [33] Jean-Luc Gauvain, Chin-Hui Lee. Bayesian learning for hidden markov model with gaussian mixture state observation densities[J]. Speech Communication, 1992, 11(2):205–213.
- [34] Douglas A Reynolds, Thomas F Quatieri, Robert B Dunn. Speaker verification using adapted gaussian mixture models[J]. Digital signal processing, 2000, 10(1):19–41.
- [35] Douglas A Reynolds, Richard C Rose. Robust text-independent speaker identification using gaussian mixture speaker models[J]. Speech and Audio Processing, IEEE Transactions on, 1995, 3(1):72–83.
- [36] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction[C]. In Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on. IEEE, 2004, volume 2, 28–31.
- [37] D-S Lee. Effective gaussian mixture learning for video background subtraction[J]. Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2005, 27(5):827–832.
- [38] Ming-Hsuan Yang, Narendra Ahuja. Gaussian mixture model for human skin color and its application in image and video databases[C]. In Proc. SPIE: Storage and Retrieval for Image and Video Databases VII. 1999, volume 3656, 458–466.
- [39] Huan Liu, Hiroshi Motoda. Computational methods of feature selection[M]. CRC Press, 2007.
- [40] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, Andrew Y Ng. Reading digits in natural images with unsupervised feature learning[C]. In NIPS workshop on deep learning and unsupervised feature learning. Granada, Spain, 2011, volume 2011, 5.
- [41] Adam Coates, Blake Carpenter, Carl Case, Sanjeev Satheesh, Bipin Suresh, Tao Wang, David J Wu, Andrew Y Ng. Text detection and character recognition in scene images with unsupervised feature learning[C]. In Document Analysis and Recognition (ICDAR), 2011 International Conference on. IEEE, 2011, 440–445.
- [42] Jennifer G Dy, Carla E Brodley. Feature selection for unsupervised learning[J]. The Journal of Machine Learning Research, 2004, 5:845–889.

- [43] Adam Coates, Andrew Y Ng. Learning feature representations with k-means. In *Neural Networks: Tricks of the Trade*, Springer, 2012, 561–580.
- [44] Jianchao Yang, Kai Yu, Yihong Gong, Thomas Huang. Linear spatial pyramid matching using sparse coding for image classification[C]. In *Computer Vision and Pattern Recognition*, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009, 1794–1801.
- [45] Y-lan Boureau, Yann L Cun, et al. Sparse feature learning for deep belief networks[C]. In *Advances in neural information processing systems*. 2008, 1185–1192.
- [46] Adam Coates, Andrew Y Ng. The importance of encoding versus training with sparse coding and vector quantization[C]. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*. 2011, 921–928.
- [47] Adam Coates, Andrew Y Ng, Honglak Lee. An analysis of single-layer networks in unsupervised feature learning[C]. In *International Conference on Artificial Intelligence and Statistics*. 2011, 215–223.
- [48] Shenghua Gao, Ivor Waihung Tsang, Liang-Tien Chia, Peilin Zhao. Local features are not lonely—laplacian sparse coding for image classification[C]. In *Computer Vision and Pattern Recognition (CVPR)*, 2010 IEEE Conference on. IEEE, 2010, 3555–3561.
- [49] Julien Mairal, Francis Bach, Jean Ponce, Guillermo Sapiro. Online learning for matrix factorization and sparse coding[J]. *The Journal of Machine Learning Research*, 2010, 11:19–60.
- [50] B-H Juang. On the hidden markov model and dynamic time warping for speech recognition—a unified view[J]. *AT&T Bell Laboratories Technical Journal*, 1984, 63(7):1213–1243.
- [51] Cory S Myers, Lawrence R Rabiner. A comparative study of several dynamic time-warping algorithms for connected-word recognition[J]. *Bell System Technical Journal*, 1981, 60(7):1389–1409.
- [52] Lawrence R Rabiner, Aaron E Rosenberg, Stephen E Levinson. Considerations in dynamic time warping algorithms for discrete word recognition[J]. *The Journal of the Acoustical Society of America*, 1978, 63(S1):S79–S79.
- [53] Donald J Berndt, James Clifford. Using dynamic time warping to find patterns in time series.[C]. In *KDD workshop*. Seattle, WA, 1994, volume 10, 359–370.
- [54] Xuedong D Huang, Yasuo Ariki, Mervyn A Jack. *Hidden Markov models for speech recognition*[M], volume 2004. Edinburgh university press Edinburgh, 1990.
- [55] Lawrence Rabiner. A tutorial on hidden markov models and selected applications in speech recognition[J]. *Proceedings of the IEEE*, 1989, 77(2):257–286.
- [56] Lawrence Rabiner, Biing-Hwang Juang. An introduction to hidden markov models[J]. *ASSP Magazine*, IEEE, 1986, 3(1):4–16.
- [57] Alex Waibel. Modular construction of time-delay neural networks for speech recognition[J]. *Neural computation*, 1989, 1(1):39–46.

- [58] Satoshi Nakamura, Kiyohiro Shikano. Speaker adaptation applied to hmm and neural networks[C]. In Acoustics, Speech, and Signal Processing, 1989. ICASSP-89., 1989 International Conference on. IEEE, 1989, 89–92.
- [59] Matthew D Zeiler, Rob Fergus. Visualizing and understanding convolutional networks. In Computer Vision–ECCV 2014, Springer, 2014, 818–833.
- [60] Matthew D Zeiler, Rob Fergus. Stochastic pooling for regularization of deep convolutional neural networks[J]. arXiv preprint arXiv:1301.3557, 2013.
- [61] Jiquan Ngiam, Zhenghao Chen, Daniel Chia, Pang W Koh, Quoc V Le, Andrew Y Ng. Tiled convolutional neural networks[C]. In Advances in Neural Information Processing Systems. 2010, 1279–1287.
- [62] Bertram E Shi. Gabor-type filtering in space and time with cellular neural networks[J]. Circuits and Systems I: Fundamental Theory and Applications, IEEE Transactions on, 1998, 45(2):121–132.
- [63] Yoshua Bengio, Yann LeCun, et al. Scaling learning algorithms towards ai[J]. Large-scale kernel machines, 2007, 34(5).
- [64] Yoshua Bengio. Learning deep architectures for ai[J]. Foundations and trends® in Machine Learning, 2009, 2(1):1–127.
- [65] Bernhard Schölkopf, Christopher JC Burges, Alexander J Smola. Advances in kernel methods: support vector learning[M]. MIT press, 1999.
- [66] Bernhard E Boser, Isabelle M Guyon, Vladimir N Vapnik. A training algorithm for optimal margin classifiers[C]. In Proceedings of the fifth annual workshop on Computational learning theory. ACM, 1992, 144–152.
- [67] Corinna Cortes, Vladimir Vapnik. Support-vector networks[J]. Machine learning, 1995, 20(3):273–297.
- [68] Kurt Hornik, Maxwell Stinchcombe, Halbert White. Multilayer feedforward networks are universal approximators[J]. Neural networks, 1989, 2(5):359–366.
- [69] Robert Hecht-Nielsen. Theory of the backpropagation neural network[C]. In Neural Networks, 1989. IJCNN., International Joint Conference on. IEEE, 1989, 593–605.
- [70] AN Komolgorov. On the representation of continuous functions of several variables by superposition of continuous functions of a smaller number of variables[C]. In Dokl. Akad. 1956, volume 108, 179–182.
- [71] Warren S McCulloch, Walter Pitts. A logical calculus of the ideas immanent in nervous activity[J]. The bulletin of mathematical biophysics, 1943, 5(4):115–133.
- [72] Donald Olding Hebb. The organization of behavior: A neuropsychological approach[M]. John Wiley & Sons, 1949.
- [73] Saburo Ikeda, Mikiko Ochiai, Yoshikazu Sawaragi. Sequential gmdh algorithm and its application to river flow prediction[J]. Systems, Man and Cybernetics, IEEE Transactions on, 1976, (7):473–479.
- [74] Stanley J Farlow. Self-organizing methods in modeling: GMDH type algorithms[M], volume 54. CrC

- Press, 1984.
- [75] Hema Rao Madala, Alexy G Ivakhnenko. Inductive learning algorithms for complex systems modeling[M]. cRc press Boca Raton, 1994.
- [76] Pavel Kordík, Pavel Náplava, Miroslav Snorek, Marko Genyk-Berezovskyj. Modified gmdh method and models quality evaluation by visualization[J]. Control Systems and Computers, 2003, 2:68–75.
- [77] Marcin Witczak, Józef Korbicz, Marcin Mrugalski, Ron J Patton. A gmdh neural network-based approach to robust fault diagnosis: Application to the damadics benchmark problem[J]. Control Engineering Practice, 2006, 14(6):671–683.
- [78] Tadashi Kondo, Junji Ueno. Multi-layered gmdh-type neural network self-selecting optimum neural network architecture and its application to 3-dimensional medical image recognition of blood vessels[J]. International Journal of innovative computing, information and control, 2008, 4(1):175–187.
- [79] Kunihiro Fukushima. Neural network model for a mechanism of pattern recognition unaffected by shift in position- neocognitron[J]. ELECTRON. & COMMUN. JAPAN, 1979, 62(10):11–18.
- [80] Gerald Tesauro. Practical issues in temporal difference learning[M]. Springer, 1992.
- [81] Vvera Kuurkova. Kolmogorov's theorem and multilayer neural networks[J]. Neural networks, 1992, 5(3):501–506.
- [82] Kurt Hornik. Approximation capabilities of multilayer feedforward networks[J]. Neural networks, 1991, 4(2):251–257.
- [83] Yann LeCun, Yoshua Bengio. Convolutional networks for images, speech, and time series[J]. The handbook of brain theory and neural networks, 1995, 3361:310.
- [84] Alex Krizhevsky, Ilya Sutskever, Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks[C]. In Advances in neural information processing systems. 2012, 1097–1105.
- [85] Kai Yu, Yuanqing Lin, John Lafferty. Learning image representations from the pixel level via hierarchical sparse coding[C]. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on. IEEE, 2011, 1713–1720.
- [86] Matthew D Zeiler, Dilip Krishnan, Graham W Taylor, Robert Fergus. Deconvolutional networks[C]. In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010, 2528–2535.
- [87] Ruslan Salakhutdinov, Geoffrey E Hinton. Deep boltzmann machines[C]. In International Conference on Artificial Intelligence and Statistics. 2009, 448–455.
- [88] Nitish Srivastava, Ruslan R Salakhutdinov. Multimodal learning with deep boltzmann machines[C]. In Advances in neural information processing systems. 2012, 2222–2230.
- [89] Ian Goodfellow, Mehdi Mirza, Aaron Courville, Yoshua Bengio. Multi-prediction deep boltzmann machines[C]. In Advances in Neural Information Processing Systems. 2013, 548–556.
- [90] Jonas Gehring, Yajie Miao, Florian Metze, Alex Waibel. Extracting deep bottleneck features using stacked

- auto-encoders[C]. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, 3377–3381.
- [91] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion[J]. *The Journal of Machine Learning Research*, 2010, 11:3371–3408.
- [92] Pascal Vincent, Hugo Larochelle, Yoshua Bengio, Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders[C]. In *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, 1096–1103.
- [93] Yann LeCun, M Ranzato. Deep learning tutorial[C]. In *Tutorials in International Conference on Machine Learning (ICML'13)*. Citeseer, 2013.
- [94] A Griewank. *Documenta mathematica-extra volume ismp*[J]. 2012.
- [95] Stephan W Director, Ronald A Rohrer. The generalized adjoint network and network sensitivities[J]. *Circuit Theory, IEEE Transactions on*, 1969, 16(3):318–323.
- [96] GW Gray, SG Wilkinson. The effect of ethylenediaminetetra-acetic acid on the cell walls of some gram-negative bacteria[J]. *Journal of general microbiology*, 1965, 39(3):385–399.
- [97] Richard Ernest Bellman, Stuart E Dreyfus. *Applied dynamic programming*[J]. 1962.
- [98] James E Kelley, Jr. The cutting-plane method for solving convex programs[J]. *Journal of the Society for Industrial & Applied Mathematics*, 1960, 8(4):703–712.
- [99] AE Bryson, RWF Gross. Diffraction of strong shocks by cones, cylinders, and spheres[J]. *Journal of Fluid Mechanics*, 1961, 10(01):1–16.
- [100] Jacques Hadamard. *Mémoire sur le problème d'analyse relatif à l'équilibre des plaques élastiques encastrées*[M], volume 33. Imprimerie nationale, 1908.
- [101] ATC Goh. Back-propagation neural networks for modeling complex systems[J]. *Artificial Intelligence in Engineering*, 1995, 9(3):143–151.
- [102] S Linnainmaa. The representation of the cumulative rounding error of an algorithm as a taylor expansion of the local rounding errors[J]. *Master's Thesis (in Finnish), Univ. Helsinki*, 1970, 6–7.
- [103] Bert Speelpenning. Compiling fast partial derivatives of functions given by algorithms. Technical report, Illinois Univ., Urbana (USA). Dept. of Computer Science, 1980.
- [104] Jürgen Schmidhuber. Deep learning in neural networks: An overview[J]. *Neural Networks*, 2015, 61:85–117.
- [105] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.
- [106] Scott E Fahlman. An empirical study of learning speed in back-propagation networks[J]. 1988.
- [107] Dilip Sarkar. Methods to speed up error back-propagation learning algorithm[J]. *ACM Computing Surveys*

- (CSUR), 1995, 27(4):519–544.
- [108] Alan S Lapedes, Robert M Farber. How neural nets work[C]. In Neural information processing systems. 1988, 442–456.
- [109] Thomas P Vogl, JK Mangis, AK Rigler, WT Zink, DL Alkon. Accelerating the convergence of the back-propagation method[J]. Biological cybernetics, 1988, 59(4-5):257–263.
- [110] Yann LeCun, Patrice Y Simard, Barak Pearlmutter. Automatic learning rate maximization by on-line estimation of the hessian’ s eigenvectors[J]. Advances in neural information processing systems, 1993, 5:156–163.
- [111] Robert A Jacobs. Increased rates of convergence through learning rate adaptation[J]. Neural networks, 1988, 1(4):295–307.
- [112] L Almeida, F Silva. Speeding up backpropagation[J]. Adv Neural Comput, 1990, 151–158.
- [113] Ralph Neuneier, Hans Georg Zimmermann. How to train neural networks. In Neural Networks: Tricks of the Trade, Springer, 1998, 373–423.
- [114] Stephen José Hanson, Lorien Y Pratt. Comparing biases for minimal network construction with back-propagation[C]. In Advances in neural information processing systems. 1989, 177–185.
- [115] Andreas S Weigend, David E Rumelhart, Bernardo A Huberman. Generalization by weight-elimination applied to currency exchange rate prediction[C]. In Neural Networks, 1991., IJCNN-91-Seattle International Joint Conference on. IEEE, 1991, volume 1, 837–841.
- [116] Stuart Geman, Elie Bienenstock, René Doursat. Neural networks and the bias/variance dilemma[J]. Neural computation, 1992, 4(1):1–58.
- [117] Geoffrey E Hinton, Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights[C]. In Proceedings of the sixth annual conference on Computational learning theory. ACM, 1993, 5–13.
- [118] Trevor J Hastie, Robert J Tibshirani. Generalized additive models[M], volume 43. CRC Press, 1990.
- [119] Trevor Hastie, Robert Tibshirani, Jerome Friedman, T Hastie, J Friedman, R Tibshirani. The elements of statistical learning[M], volume 2. Springer, 2009.
- [120] Frederick Mosteller, John W Tukey. Data analysis, including statistics[J]. 1968.
- [121] David Mackay. The evidence framework applied to classification networks[J]. Neural computation, 1992, 4(5):720–736.
- [122] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition[J]. Neural computation, 1989, 1(4):541–551.
- [123] Pierre Baldi, Yves Chauvin. Neural networks for fingerprint recognition[J]. Neural Computation, 1993, 5(3):402–418.

- [124] Steve Lawrence, C Lee Giles, Ah Chung Tsoi, Andrew D Back. Face recognition: A convolutional neural-network approach[J]. *Neural Networks, IEEE Transactions on*, 1997, 8(1):98–113.
- [125] Ossama Abdel-Hamid, Abdel-rahman Mohamed, Hui Jiang, Gerald Penn. Applying convolutional neural networks concepts to hybrid nn-hmm model for speech recognition[C]. In *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, 4277–4280.
- [126] Claus Nebauer. Evaluation of convolutional neural networks for visual recognition[J]. *Neural Networks, IEEE Transactions on*, 1998, 9(4):685–696.
- [127] Srinivas C Turaga, Joseph F Murray, Viren Jain, Fabian Roth, Moritz Helmstaedter, Kevin Briggman, Winfried Denk, H Sebastian Seung. Convolutional networks can learn to generate affinity graphs for image segmentation[J]. *Neural Computation*, 2010, 22(2):511–538.
- [128] Pierre Sermanet, Yann LeCun. Traffic sign recognition with multi-scale convolutional networks[C]. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011, 2809–2813.
- [129] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, Christian Igel. The german traffic sign recognition benchmark: a multi-class classification competition[C]. In *Neural Networks (IJCNN), The 2011 International Joint Conference on*. IEEE, 2011, 1453–1460.
- [130] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, Li Fei-Fei. Imagenet: A large-scale hierarchical image database[C]. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, 2009, 248–255.
- [131] Yaniv Taigman, Ming Yang, Marc’Aurelio Ranzato, Lior Wolf. Deepface: Closing the gap to human-level performance in face verification[C]. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, 1701–1708.
- [132] Gary B Huang, Manu Ramesh, Tamara Berg, Erik Learned-Miller. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Technical report, Technical Report 07-49, University of Massachusetts, Amherst, 2007.
- [133] Karen Simonyan, Andrew Zisserman. Two-stream convolutional networks for action recognition in videos[C]. In *Advances in Neural Information Processing Systems*. 2014, 568–576.
- [134] Alexander Toshev, Christian Szegedy. Deeppose: Human pose estimation via deep neural networks[C]. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014, 1653–1660.
- [135] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, Martin Riedmiller. Playing atari with deep reinforcement learning[J]. *arXiv preprint arXiv:1312.5602*, 2013.
- [136] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio. Generative adversarial nets[C]. In *Advances in Neural Information Processing Systems*. 2014, 2672–2680.

- [137] Charles F Cadieu, Ha Hong, Daniel LK Yamins, Nicolas Pinto, Diego Ardila, Ethan A Solomon, Najib J Majaj, James J DiCarlo. Deep neural networks rival the representation of primate it cortex for core visual object recognition[J]. *PLoS computational biology*, 2014, 10(12):e1003963.
- [138] Orhan Firat, Emre Aksan, Ilke Oztekin, Fatos T Yarman Vural. Learning deep temporal representations for brain decoding[J]. *arXiv preprint arXiv:1412.7522*, 2014.
- [139] Thorsten Barnickel, Jason Weston, Ronan Collobert, Hans-Werner Mewes, Volker Stümpflen. Large scale application of neural network based semantic role labeling for automated relation extraction from biomedical texts[J]. *PLoS One*, 2009, 4(7):e6393.
- [140] Ronan Collobert, Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning[C]. In *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, 160–167.
- [141] Yann LeCun, Corinna Cortes. The mnist database of handwritten digits[EB/OL], 1998.
- [142] A Nemirovski, D Yudin. On cezari’ s convergence of the steepest descent method for approximating saddle point of convex-concave functions[C]. In *Soviet Math. Dokl.* 1978, volume 19.
- [143] Martin Zinkevich. Online convex programming and generalized infinitesimal gradient ascent[J]. 2003.
- [144] Benjamin Blankertz, K Muller, Dean J Krusienski, Gerwin Schalk, Jonathan R Wolpaw, Alois Schlogl, Gert Pfurtscheller, Jd R Millan, M Schroder, Niels Birbaumer. The bci competition iii: Validating alternative approaches to actual bci problems[J]. *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, 2006, 14(2):153–159.
- [145] F Sharbrough, GE Chatrian, RP Lesser, H Lüders, M Nuwer, TW Picton. American electroencephalographic society guidelines for standard electrode position nomenclature[J]. *J. clin. Neurophysiol*, 1991, 8(2):200–202.
- [146] Alain Rakotomamonjy, Vincent Guigue. Bci competition iii: dataset ii-ensemble of svms for bci p300 speller[J]. *Biomedical Engineering, IEEE Transactions on*, 2008, 55(3):1147–1154.
- [147] Benjamin Blankertz, Gabriel Curio, Klaus-Robert Muller. Classifying single trial eeg: Towards brain computer interfacing[J]. *Advances in neural information processing systems*, 2002, 1:157–164.
- [148] Soroosh Solhjoo, Ali Motie Nasrabadi, Mohammad Reza Hashemi Golpayegani. Classification of chaotic signals using hmm classifiers: Eeg-based mental task classification[C]. In *Proceedings of the European Signal Processing Conference*. 2005.
- [149] Silvia Chiappa, Samy Bengio. Hmm and iohmm modeling of eeg rhythms for asynchronous bci systems. Technical report, IDIAP, 2003.
- [150] Shi Zhong, Joydeep Ghosh. Hmms and coupled hmms for multi-channel eeg classification[C]. In *Proceedings of the IEEE International Joint Conference on Neural Networks*. 2002, volume 2, 1254–1159.
- [151] Torsten Felzer, B Freisieben. Analyzing eeg signals using the probability estimating guarded neural clas-

- sifier[J]. *Neural Systems and Rehabilitation Engineering*, IEEE Transactions on, 2003, 11(4):361–371.
- [152] Charles W Anderson, Saikumar V Devulapalli, Erik A Stolz. Determining mental state from eeg signals using parallel implementations of neural networks[J]. *Scientific programming*, 1995, 4(3):171–183.
- [153] Hubert Cecotti, Axel Gräser. Time delay neural network with fourier transform for multiple channel detection of steady-state visual evoked potential for brain-computer interfaces[C]. In *Proc. of European Signal Processing Conference*. 2008.
- [154] Nikola Masic, Gert Pfurtscheller, Doris Flotzinger. Neural network-based predictions of hand movements using simulated and real eeg data[J]. *Neurocomputing*, 1995, 7(3):259–274.
- [155] Nikola Masic, Gert Pfurtscheller. Neural network based classification of single-trial eeg data[J]. *Artificial Intelligence in Medicine*, 1993, 5(6):503–513.
- [156] Hubert Cecotti, Axel Graser. Convolutional neural networks for p300 detection with application to brain-computer interfaces[J]. *Pattern Analysis and Machine Intelligence*, IEEE Transactions on, 2011, 33(3):433–445.
- [157] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding[J]. *arXiv preprint arXiv:1408.5093*, 2014.
- [158] Ossama Abdel-Hamid, Abdel-Rahman Mohamed, Hui Jiang, Li Deng, Gerald Penn, Dong Yu. Convolutional neural networks for speech recognition[J]. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 2014, 22(10):1533–1545.
- [159] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, Jürgen Schmidhuber. Convolutional neural network committees for handwritten character classification[C]. In *Document Analysis and Recognition (ICDAR)*, 2011 International Conference on. IEEE, 2011, 1135–1139.
- [160] Sean R Eddy. Hidden markov models[J]. *Current opinion in structural biology*, 1996, 6(3):361–365.
- [161] Hava T Siegelmann, Eduardo D Sontag. On the computational power of neural nets[J]. *Journal of computer and system sciences*, 1995, 50(1):132–150.
- [162] Barak A Pearlmutter. Gradient calculations for dynamic recurrent neural networks: A survey[J]. *Neural Networks*, IEEE Transactions on, 1995, 6(5):1212–1228.
- [163] Yoshua Bengio, Paolo Frasconi. An input output hmm architecture[J]. *Advances in neural information processing systems*, 1995, 427–434.
- [164] Sepp Hochreiter. Untersuchungen zudynamischenneuronallennetzen[J]. Master’s thesis, Institut für Informatik, Technische Universität, München, 1991.
- [165] Sepp Hochreiter, Jürgen Schmidhuber. Long short-term memory[J]. *Neural computation*, 1997, 9(8):1735–1780.
- [166] Felix A Gers, Jürgen Schmidhuber, Fred Cummins. Learning to forget: Continual prediction with lstm[J].

- Neural computation, 2000, 12(10):2451–2471.
- [167] Ronald J Williams, Jing Peng. An efficient gradient-based algorithm for on-line training of recurrent network trajectories[J]. Neural computation, 1990, 2(4):490–501.
- [168] AJ Robinson, Frank Fallside. The utility driven dynamic error propagation network[M]. University of Cambridge Department of Engineering, 1987.
- [169] Ronald J Williams, David Zipser. A learning algorithm for continually running fully recurrent neural networks[J]. Neural computation, 1989, 1(2):270–280.
- [170] Jürgen Schmidhuber. A fixed size storage $O(n^3)$ time complexity learning algorithm for fully recurrent continually running networks[J]. Neural Computation, 1992, 4(2):243–248.
- [171] Ronald J Williams, David Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity[J]. Back-propagation: Theory, architectures and applications, 1995, 433–486.
- [172] Yoshua Bengio, Patrice Simard, Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult[J]. Neural Networks, IEEE Transactions on, 1994, 5(2):157–166.
- [173] Sepp Hochreiter, Yoshua Bengio, Paolo Frasconi, Jürgen Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies[EB/OL], 2001.
- [174] Nicole Beringer, Alex Graves, Florian Schiel, Jürgen Schmidhuber. Classifying unprompted speech by retraining lstm nets. In Artificial Neural Networks: Biological Inspirations–ICANN 2005, Springer, 2005, 575–581.
- [175] Hasim Sak, Andrew Senior, Françoise Beaufays. Long short-term memory recurrent neural network architectures for large scale acoustic modeling[C]. In Proceedings of the Annual Conference of International Speech Communication Association (INTERSPEECH). 2014.
- [176] Ilya Sutskever, Oriol Vinyals, Quoc VV Le. Sequence to sequence learning with neural networks[C]. In Advances in Neural Information Processing Systems. 2014, 3104–3112.
- [177] Alex Graves. Connectionist temporal classification. In Supervised Sequence Labelling with Recurrent Neural Networks, Springer, 2012, 61–93.
- [178] Théodore Bluche, Jérôme Louradour, Maxime Knibbe, Bastien Moysset, Mohamed Faouzi Benzeghiba, Christopher Kermorvant. The a2ia arabic handwritten text recognition system at the open hart2013 evaluation[C]. In Document Analysis Systems (DAS), 2014 11th IAPR International Workshop on. IEEE, 2014, 161–165.
- [179] Alex Graves, Santiago Fernández, Faustino Gomez, Jürgen Schmidhuber. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks[C]. In Proceedings of the 23rd international conference on Machine learning. ACM, 2006, 369–376.
- [180] Douglas Eck, Juergen Schmidhuber. A first look at music composition using lstm recurrent neural networks[J]. Istituto Dalle Molle Di Studi Sull'Intelligenza Artificiale, 2002.

- [181] Sepp Hochreiter, Martin Heusel, Klaus Obermayer. Fast model-based protein homology detection without alignment[J]. Bioinformatics, 2007, 23(14):1728–1736.
- [182] Yoshua Bengio, Nicolas Boulanger-Lewandowski, Razvan Pascanu. Advances in optimizing recurrent networks[C]. In Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on. IEEE, 2013, 8624–8628.

致谢

在我写这个文档的过程中, 得到了网络上很多网贴的帮助, 在此感谢 baidu, Google, 感谢 CTeX 社区 <http://www.ctex.org>, L^AT_EX 学习园地: <http://blog.sina.com.cn/wangzhaoli11>, 中科大 CTAN 镜像 <http://mirrors.ustc.edu.cn/CTAN/>, 水木社区 T_EX 版等网站、论坛, 其他一些较小的个人网站, 论坛不再一一点名, 在此一并感谢。感谢浙江大学数学系提供的原始模版, 感谢 88T_EX 版。