上海交通大学硕士学位论文

基于卷积神经网络的深度学习算法研究与实现

硕士研究生：
学　　号：
导　　师：
副 导 师：
学　　科：集成电路工程
所在单位：电子信息与电气工程学院
答辩日期：2017 年 01 月
授予学位单位：上海交通大学

Dissertation Submitted to Shanghai Jiao Tong University
for the Degree of Master

DESIGN AND IMPLEMENTATION OF DEEP LEARNING ALGORITHM BASED ON CONVOLUTIONAL NEURAL NETWORK

Candidate：

Student ID:

Supervisor：

Assistant Supervisor：

Speciality： Integrated Circuit Engineering

Affiliation： School of Electronic Information and Electrical Engineering

Date of Defence： Jan,2017

Degree-Conferring-Institution： Shanghai Jiao Tong University

基于卷积神经网络的深度学习算法研究与实现

摘 要

随着芯片处理性能的增长、机器学习算法的进步以及数据的爆发式增长，深度学习等相关领域算法成了国内外研究的热点。卷积神经网络（Convolutional Neural Network，CNN）在计算机视觉领域取得了巨大的突破和进展。然而其性能的提升大多都依赖于 CNN 网络模型层数的加深和训练数据集的扩充，尤其是卷积层数的加深会造成模型复杂度急剧上升。因此，设计具有低复杂度的 CNN 模型变得至关重要。

针对 CNN 模型复杂度高的问题，本文首先利用五层的 CNN 模型，应用于人脸识别的场景中。为了提高识别的准确率，设计了一种深度 CNN 模型，加深了网络的卷积层数和全连接层数，增加模型的特征图数量和全连接层维数。为了降低复杂度，设计了一种轻量 CNN 模型。通过使用最大特征映射（Max Feature Map，MFM）激活函数来替代修正线性单元（Rectified Linear Unit，ReLU），使得特征图数量每经过一次激活层都会减半，参数规模也会减少一半，在激活层降低了模型的复杂度；进一步，通过引入嵌套网络（Network in Network，NIN），在网络中有选择地使用尺寸为 1×1 的卷积核对特征图进行卷积操作，使得该卷积层的参数减少为原来的 1/9。在网络层数不变的情况下，也增强模型的特征抽象能力。同时，降低网络模型全连接层的维度能够进一步降低模型参数数量，在尽量保证准确率的情况下最大程度上降低模型的复杂度。

测试结果表明：单独通过添加卷积层的层数，能够提升人脸识别的复杂度，深度 CNN 模型相比 CNN 模型准确率提升了 15%，但与此同时，前者的模型尺寸是后者的 28 倍之多。通过对深度 CNN 模型进行复杂度优化，轻量 CNN 模型相比 CNN 模型准确率提升 5%，尺寸

只有深度 CNN 模型的 5.6%。轻量 CNN 模型在复杂度方面也要优于文献中的 GoogLeNet、ResNet 等网络模型，验证了轻量 CNN 模型在降低模型复杂度方面的有效性。

关键词：深度学习；卷积神经网络；人脸识别；最大特征映射；嵌套网络

DESIGN AND IMPLEMENTATION OF DEEP LEARNING ALGORITHM BASED ON CONVOLUTIONAL NEURAL NETWORK

ABSTRACT

With the development of SoC processing performance, the progress of machine learning algorithm and the explosive increase of data, deep learning has drawn more and more attention. Convolutional Neural Network(CNN) has made great breakthrough and progress in the field of computer vision. However, most of the performance improvement depend on the increase of the convolutional layer and the expansion of the training data set, which could make complexity rise sharply. Therefore, it's important to design the CNN model with low complexity.

To address this issue, a CNN model with five convolutional layers is designed to be applied to face recognition problem. In order to improve the accuracy, we add more the convolutional layers and full connection layers, increase the feature maps and full connection dimensions. To reduce the complexity, max feature map activation function is utilized to replace the rectified linear unit function. Each time the feature maps go through the activation layer, the parameters could be reduced by half in the activation layer. Moreover, network in network is introduced in the CNN model. In this network, the convolution operation is performed by a filter of size 1×1 so that the parameters of the convolution layer could be reduced by 8/9. Meanwhile, it will also enhance the generalization ability without change of the number of layers. In addition, the dimension of full connection lays is decreased to further reduce the parameters as far as possible to ensure the accuracy.

Test results demonstrate that the accuracy of deep CNN model increases by 15% compared to CNN model by adding convolutional layers simply. However, the size of deep CNN model is 28 times bigger than the CNN one. After the complexity optimization of the deep CNN model, the accuracy of light CNN model increases by 5% compared to CNN model

and the size of light CNN model is 1/20 the size of deep CNN model. In the meantime, light CNN model outperforms the GoogLeNet and ResNet models in terms of complexity. The test results verify the effectiveness of light CNN model in reducing the model complexity.

KEY WORDS: deep learning; convolutional neural network; face recognition; max feature map; network in network

目 录

图 录

表 录

第一章 绪论

1.1 论文研究的背景

深度学习（Deep Learning）起源于神经网络的研究，对于神经网络而言，深度指的是网络学习得到的函数中非线性运算组合水平的层数，当前神经网络的学习算法多是针对较浅层次的网络结构，深度结构神经网络则拥有较多的隐层。目前的很多机器学习算法都是基于浅层结构算法，它们在很多方面存在一定的局限性，比如人工选择特征、在样本有限的情况下表达复杂函数的能力有限，且针对复杂的分类问题，浅层学习的泛化能力受到一定制约。而深度学习算法是一种深层非线性网络结构，尝试通过使用具有多个处理层的神经网络来对数据中的高级抽象特征进行建模。相比于浅层学习而言，深度学习能够较好地表征表达复杂目标函数。当网络规模增大时，浅层学习需要增加指数级的训练参数，深度学习能够紧凑地表达这一网络结构，降低计算的复杂度。

卷积神经网络（Convolutional Neural Network，CNN）是最典型的深度学习模型，在计算机视觉领域已经成为最流行的技术之一。CNN 是由传统神经网络结构优化而来，它基于神经元之间的局部连接和层次结构组织图像转换，将有相同参数的神经元应用于前一层神经网络的小块区域时，得到一种相对于数据平移保持不变的神经网络结构形式。CNN 结构特别适用于处理二维数据，即图像数据的处理。在网络结构中，图像中的分块数据，即共享权值的局部感知区域，前向传播通过网络中紧密联系的各个过滤层，挖掘到图像数据中最能表征图像特性的特征，同时这种特征对于位移、旋转和拉伸具有一定程度的不变性。CNN 结构能够高效地自动提取图像的特征用于各种应用场景，这一优势是传统的基于像素和自选特征等算法所不能比较的。许多实际场景任务诸如图像分类、物体检测、人脸识别等的提升都受益于卷积神经网络模型，其中的一些已经运用于商用系统。

由于 CNN 可以从大规模训练数据集中学习复杂数据的分布，因此在计算机视觉领域取得了巨大的突破和进展。2014 年，香港中文大学的深度卷积神经网络结构 DeepID[1]在 LFW[2]数据库中的人脸识别正确率达到了 97.45%。随后，通过不断加深网络结构和不断增大训练数据集的规模等方法[3-6]又将人脸识别的准确率提升到了 99%，已然超过人的识别正确率。然而随着网络结构的加深和训练集规模不断增大，模型的复杂度不断增加，会导致训练时间和识别时间的急剧增加，这也成为制约 CNN 模型推广应用的关键。因此，直接训练准确率高，复杂度低的 CNN 模型非常重要。

基于以上的背景，本文主要对适用于人脸识别的低复杂度卷积神经网络模型

进行了学习与研究。本文尝试从卷积层和激活层两个方面，对网络模型进行优化。在尽量保证准确率的情况下，降低模型的复杂度。

1.2 国内外研究现状

1.2.1 深度学习的研究

近些年来，随着芯片处理性能的增长、计算成本的降低、机器学习算法的显著进步[7]以及数据的爆发式增长，深度学习等相关领域算法成了国内外研究的热点。深度学习是用于学习数据表示的更广泛的机器学习领域的一部分，通常使用许多层的非线性处理单元的级联来进行特征提取和变换。每个连续层使用来自上一层的输出作为输入，形成从低级特征到高级特征的层级[8]。

深度学习概念是在 1986 由 Rina Dechter 首次提出[9]，之后在 2000 年 Igor Aizenberg 等人引入了人工神经网络[10]。但是，随着神经网络深度的增加，会出现梯度消失现象，即非凸目标函数产生局部最优解，这种现象随着网络深度的增加而恶化，阻止深度网络算法那达到全局最优解。2006 年，Geoffrey Hinton 等人提出的深度置信网络(Deep Belief Network)的无监督高效学习算法[11]，解决了深度学习模型优化的难题，通过显示多层前馈神经网络可以如何有效地预训练每一层，依次处理每层作为一个无监督受限玻尔兹曼机，然后再使用监督反向传播算法进行调整[12]。

自此之后，大量学术研究人员对深度学习算法进行了深入的研究，极大推动了深度学习理论的发展；同时经过近几年的沉淀，深度学习的模型已被应用到语音识别、计算机视觉、自然语言处理等领域，并取得了良好的效果。

在语音识别领域中，神经网络在语音识别方面有很长的历史，通常与隐马尔科夫模型结合[13]。 近年来，由于深层前馈网络产生的声学模型的显著改进，深度学习在语音识别的应用已经获得了极大的关注[14]。考虑到语音是一个固有的动态过程，所以递归神经网络（Recurrent Neural Network，RNN）是一种非常合适的网络模型。隐马尔科夫模型（Hidden Markov Model，HMM）与 RNN 结合的识别系统[15]表现也很出色，在不同噪声环境下，都取得了很好的识别率。深度学习使用端到端（End-to-End）的方式训练 RNN 用于语音识别[16]。与 HMM 相比，该方法利用 RNN 的更大的状态空间和更丰富的动态，并且避免了使用可能不正确的比对作为训练目标的问题。文献[17]研究深度递归神经网络，它结合了多层次的表示和长短期记忆（Long Short Term Memory，LSTM）。当采用端到端的方式训练模型并进行适当的正则化，深度 LSTM-RNN 实现了在语音识别基准测试集错误率 17.7％。

在计算机视觉领域中，深度学习已经应用到了诸多计算机视觉的方向，比如

图像分类、物体检测、图像检索、图像分割和姿态估计。在 2012 年的大规模视觉识别挑战赛（ILSVRC）中，Krizhevsky[18]等人提出的基于 CNN 的 AlexNet 在比赛中获得了 15.3%的 top-5 误差率，取得了当时的第一名。之后的几年里，基于 CNN 的深度网络不断刷新识别的准确率，目前已经超过人类的识别准确率。文献[19]提出了基于 CNN 的物体检测方法，把高容量卷积神经网络应用到自下而上的区域提取中，以便定位和分割对象，达到了 53.3%平均检测准确率，相比之前提升了 30%。在图像检索中，文献[20]使用 CNN 来生成二进制哈希码，用于快速的图像检索，在 CIFAR-10 数据集测试达到了 89%的准确率。在其他计算机视觉的方向中，基于深度学习框架的算法也在不断提升原有方法的性能和效率。

在自然语言处理领域中，从 2013 年开始，深度学习才在自然语言处理领域发挥作用。Mikolov 等人提出的 word2vec 模型[21]通过训练将每个词映射成 K 维实数向量，再通过词之间的距离来判断它们之间的语义相似度。2014 年的热点是各种新颖的词表示学习方法，而 2015 年则扩展到句子层次，CNN、RNN、LSTM 等模型在机器翻译、文档摘要、阅读理解、关系抽取等任务上都取得了重要进展。

1.2.2 卷积神经网络的研究

研究 CNN 的灵感来自生物的自然视觉感知机制。在 1959 年，Hubel 和 Wiesel [22]发现动物视觉皮层中的细胞负责检测接受场中的光。受到这一发现的启发，Kunihiko Fukushima 在 1980 年提出了神经认知机[23]，这可以被认为是 CNN 的前身。1990 年，LeCun 等人[24]发表了建立 CNN 现在框架的创新论文，后来又对其进行了改进[25]。他们开发了一个称为 LeNet-5 的多层人工神经网络，可以对手写数字进行分类。像其他神经网络一样，LeNet-5 有多层，可以用反向传播（Backpropagation，BP）算法训练[26]。它可以获得原始图像的有效表示，这使得它有可能直接从原始像素识别视觉模式，而且几乎没有对数据的预处理。然而，由于当时缺乏大的训练数据和计算能力，LeNet-5 在更复杂的问题（例如大规模图像和视频分类）上不能很好地执行。

自 2006 年以来，已经有许多方法来克服在训练深度 CNN 中遇到的困难。最值得注意的是，Krizhevsky 等人提出了一个经典的 CNN 架构，相比于以前的方法，这个框架显示出对图像分类任务的重大改进。他们方法的总体架构，即 AlexNet[18]，类似于 LeNet-5 但具有更深的结构。随着 AlexNet 的成功，相继有几个网络框架被提出来提高其性能。其中，四个代表性模型是 ZFNet[27]，VGGNet[28]，GoogleNet[29]和 ResNet[30]。从以上架构的演变来看，一个典型的趋势是网络深度逐渐增加，例如，赢得 2015 年 ILSVRC 冠军的 ResNet，比 AlexNet 深 20 倍，比 VGGNet 深 8 倍。通过增加深度，网络可以在不断增加的非线性中近似目标函数，并获得更好的特征表示。然而，它也极大增加了网络的复杂性，

这使得网络更难以优化并且更容易过度拟合。

近几年 CNN 也逐渐应用到人脸识别的应用场景中，其中准确率和复杂度是最重要的评价指标。DeepFace[31]是第一个训练 CNN 用于人脸验证的网络结构，通过 440 万张图像来训练 CNN 模型，特征表示层产生 4096 维的向量用于人脸验证，在 LFW 数据集上达到 97.35％的精度。作为 DeepFace 的扩展，Web-Scale[32]应用 bootstrap 方法从大型数据集中选择有效的数据作为训练集，进一步提升了准确率。文献[1][6]采用了多局部块集成模型。在不同的局部块上训练 25 个 CNN 模型，并且应用联合贝叶斯方法来获得鲁棒的特征空间。与 DeepFace 相比，他们没有使用 3D 脸部对齐。在 20 万人脸图像的数据集中训练了多个 CNN 模型，他们最终在 LFW 上获得 99.47％的准确率[33]。

在复杂度方面，目的是确定一个具有很少参数但能保持相当精度的 CNN 模型。文献[34]提出直接对预训练的 CNN 模型应用奇异值分解（Singular Value Decomposition，SVD）来降低特征维数。Han 等人提出了网络剪枝策略[35]，从预训练模型开始，然后用零替换低于某个阈值的参数以形成稀疏矩阵，最后再对稀疏的卷积神经网络训练。之后，Han 又通过将网络剪枝策略和哈夫曼编码相结合，提出了一种称为深度压缩的方法来对压缩 CNN 模型，实现大幅加速。Landola 等人[36]在设计了一种 Fire Model 来压缩网络参数，通过有选择地替换卷积核尺寸为 1×1，并且减少输入 3×3 卷积层的特征图数量，在特征图维度进行拼接，使模型减小了 50 倍的大小，并且还保持了相当高的准确率。

从文献中可以看出，虽然以前的 CNN 模型已经达到了在 LFW 数据集测试的极限精度，但是复杂度仍然是一个非常棘手的问题，这会成为嵌入式设备或智能手机上进行人脸识别功能的瓶颈。

首先，随着 CNN 的加深，网络结构的模型会逐渐增大，这会使得在 CPU 或 GPU 上特征提取的时间大大增加。同时当深度网络的规模逐渐变大时，所需要训练的权值非常多，因此训练一般需要较长的时间。尤其是在现今数据量爆发式增长的情况下，深度网络的训练时间成倍增加，这明显拖慢了研究的脚步；利用生成的网络模型来进行识别，在一般的 PC 机上需要几秒才能完成人脸识别的任务，这在实际的应用中体验将会很差。特别地，当图像中有多面部需要识别时，这将耗费非常多的时间，并且引入了由面部标志的自动定位引起的不确定性。

第二，目前准确率较高的 CNN 结构模型大多数是基于线性修正单元（Rectified Linear Unit，ReLU）激活函数，使得学习到的特征通常是高维的并且很稀疏，这样会导致模型冗余参数的增多。为了获得低维和紧凑的特征表示，在网络结构中需要使用联合贝叶斯[6]或度量学习[3]来减少学习的高维稀疏特征。

因此，具有较高准确率、低复杂度的卷积神经网络模型仍然亟待研究。

1.3 研究内容

针对目前存在的问题，本文主要研究基于 CNN 的深度学习算法与框架，并将 CNN 应用在典型的人脸识别问题上，同时针对人脸识别准确率和计算复杂度的问题，通过加深网络层次和优化激活函数等方式对其进行改进。主要内容包括以下几个方面：

1. 研究研究深度学习背景知识，分析和研究现有深度学习的典型算法和各自特点，尤其是在计算机视觉方面。
2. 研究卷积神经网络，包括其区别于人工神经网络的特性、卷积神经网络的结构组成。
3. 针对人脸识别问题，构建 CNN 模型，训练得到网络模型。根据网络模型对测试集的识别准确率和识别时间（复杂度），分析可能的影响因素，综合网络模型的具体情况，通过加深网络层次和优化激活函数等方式对其进行改进。
4. 对改进的网络模型进行分析和验证。分析得出的数据，说明优化方式的合理性和正确性。

1.4 论文章节安排

本文主要针对基于卷积神经网络的深度学习算法进行研究，构建不同的深度学习框架来解决人脸识别问题。本论文主要围绕着卷积神经网络的网络架构设计和激活函数的优化两方面展开，分为五个章节，其中第三章和第四章是本文的重点，具体内容安排如下：

第一章为绪论，主要介绍本文的研究背景，研究意义以及关于深度学习和卷积神经网络的发展历史和在不同领域内的应用，对卷积神经网络在计算机视觉领域的发展和应用做了比较详细的介绍。

第二章则是对卷积神经网络进行深度的剖析。主要介绍了传统的人工神经网络，讨论了卷积神经网络区别于人工神经网络的特点，主要包括局部连接、权值共享、池化操作。然后介绍了卷积神经网络的整体架构以及组成网络的各个表示层。

第三章从人脸识别应用的场景出发，首先介绍了基础的 CNN 架构模型、网络设计参数、训练集以及相关的训练流程和参数。在此基础上，给出了更深的网络模型和设计参数用于提升人脸识别的准确率。同时，还给出两种模型的性能、复杂度测试结果及分析。

第四章讨论在现有的网络模型的基础上，降低模型复杂度的方法。重点介绍

了嵌套网络和最大特征映射两种方法用于降低模型的复杂度，并基于此设计轻量卷积神经网络模型，给出了模型与深度卷积神经网络模型性能和复杂度的测试结果及分析。

第五章介绍了本文使用的深度学习框架，给出了测试集及其处理方法。对第三章和第四章的基础网络结构、深度网络结构和轻量网络结构进行对比分析。

第六章则是对全文进行总结以及后续研究的展望。归纳总结了本论文的主要工作与创新点，并对未来卷积神经网络在人脸识别应用的研究方向进行了讨论。

第二章 卷积神经网络的结构及算法

2.1 人工神经网络

人工神经网络（Artificial Neural Network，ANN）的最初构想是模拟基本生物体的中枢神经系统。ANN 由多个神经元构成，神经元与神经元之间相互连接组成神经网络，用以处理及传递消息。当外部感知节点（神经突触）接收到信号时，神经元通过激活方程将信息从一个节点传送到下一个节点，以此来模拟人脑进行学习。神经元之间的互连通过大量的训练，能够建立输入和输出之间的关系，模仿人类完成某些特定的任务。

2.1.1 感知器单元

生物神经中枢的每一个神经元，都是一个接受信息的基本单位。在神经网络中，具有相似功能的感知器单元（Perceptron Unit）是神经网络的基本单位。

1957 年 Rosenblatt 最早将单个神经元应用在字符识别方面，感知器[37]的概念也因此被提出。在神经网络中运用感知器单元来完成学习功能建立输入输出的关系，可以将一个用数字向量表示的输入进行分类，判断其是否属于某一特定的类。单一的感知器单元是就是一种线性分类器，它实现了一个分类算法，将一组输入特征矢量进行加权，使得其经过变换后的预测结果是一个线性预测函数。感知器单元的结构如图 2-1 所示。

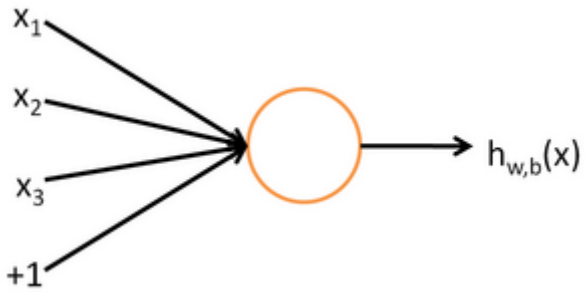


图 2-1 感知器单元

Fig.2-1 Perceptron Unit

在图 2-1 中，输入由x_1，x_2，x_3三个输入组成，+1 代表该感知器的偏置，这里记做 b。偏置 b 使得决策边界偏离原点，且不依赖于任何输入值。输入的线性组合加上偏置后，经过激活函数 f 变换，得到输出 h，即

$$h_{w,b}(x) = f(w \cdot x + b) \tag{2-1}$$

其中 $w \cdot x = \sum_{i=1}^{m} w_i \cdot x_i$，$w$ 是实值向量的权重矩阵，w 点乘 x 是原始输入向量的加权

后的输出向量，m 是感知器的输入的个数。

在使用感知器单元执行任何有监督的任务之前，必须对其进行训练，以使得其根据先验经验进行学习。在神经网络的数学模型中，预测的关键要素是边的权重 w，训练的目的就是不断调整其权值直到这个感知器能够正确分类所有的训练样本。通过神经网络学习到的知识，会以边的权重 w 和节点偏差 b 的形式存储在边和节点中。

激活函数 f 描述了输入和输出之间的关系，也称为传递函数。感知器单元通常使用单位阶跃函数作为激活函数。因此，感知算法也称为单层感知。作为线性分类器，单层感知是最简单的前馈神经网络。

举一个最简单的监督学习的例子，输入为一组带标志的向量$(x_i,\ y_i)$，$y_i \in \{0,1\}$，使用感知器单元实现二元分类的学习算法。通过函数 f 将一个实值输入向量 X 映射到唯一输出二进制实数 $f(x)$：

$$f(x)=\begin{cases}1 & \mathrm{w}\cdot x+b>0\\ 0 & \text{otherwise}\end{cases} \tag{2-2}$$

在这个二元分类问题中，$f(x)$的值（0 或 1）被用来判断点 x 的正负。如果偏置 b 为负，则输入的加权组合必须产生一个大于$|b|$的正值，才能使得输出大于分类神经元的阈值 0。需要注意的是，由于单层感知器单元是一个线性分类器，如果训练集不可线性划分，那么训练过程不会停止且永远无法得到所有向量均正确划分的分类网络。

由于多层感知器中，每一个感知器单元都独立工作，因此研究一个单位感知器的工作原理可以作为下文介绍多层感知器的基础。我们将训练次数定义为 t，t 为递增的正整数。单层感知器单元的求解过程如下：

1) 初始化权值和阈值：

权重可被初始化为 0 或接近于 0 的随机数。在这里我们使用 0，阈值设为 0。

2) 更新权重：

对于训练集中的第 j 个元素x_i及它的期望值y_i进行如下步骤的处理：

a) 计算出实际的输出：

$$\begin{aligned}h_j(t)&=f[w(t)\cdot x_j]\\&=f[w_0(t)x_{j,0}+w_1(t)x_{j,1}+\cdots+w_m(t)x_{j,m}]\end{aligned} \tag{2-3}$$

b) 更新权重：

$$w_i(t+1)=w_i(t)+(y_j-h_j(t))x_{j,i} \tag{2-4}$$

3) 迭代：

该算法经过步骤 2a 和 2b 更新权重 w 会立即应用到下组输入向量中，并

且随后继续更新，而不是等待直到训练集中的所有向量均完成计算。重复步骤直到迭代的误差小于用户指定的错误阈值或迭代预定次数已达到规定上限。

判断结果是否收敛是决定感知器是否成功学习的标准。如果训练集是线性可分，则感知器会确保达到收敛，并且达到收敛的训练次数有一个固定的上界。但是，上文提到由于感知器是线性分类器，因此如果训练集不是线性可分，那么它不会达到正确分类所有样本的状态。因此，如果训练集的先验线性可分性不可知，那么就应该它在单层感知器的基础上加以改进，能够对线性不可分数据进行分类，即下节介绍的多层感知器。

2.1.2 多层感知器

多层感知器[38]（Multi-Layer Perceptron，MLP）是由多个单层感知器组成的人工神经网络，它实现了将输入数据集合映射到一组合适的输出集合的功能。一个多层感知器为一个包含多层节点的有向图，每一层节点均完全连接到下一层。除了输入节点，每个节点都是一个具有非线性激活函数的神经元。MLP 改进了单层感知器，解决了单层感知器只能处理线性分类的弊端，可以区分线性不可分的数据。

在介绍多层感知器之前，我们先阐述神经网络的结构，如图 2-2。

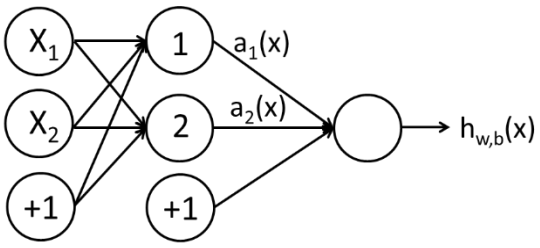


图 2-2 神经网络结构

Fig.2-2 Neural Network Structure

神经网络由多层神经元组成，最左边为输入层，最右边为输出层，中间层为隐藏层。对于第 l+1 层的节点，其输入值为第 l 层节点的输出值。以图 2-2 为例：

$$a_1(x) = f(w_1 \cdot x + b_1) \tag{2-5}$$

$$a_2(x) = f(w_2 \cdot x + b_2) \tag{2-6}$$

$$h_{w,b}(x) = f(w_h \cdot x + b_h) \tag{2-7}$$

其中 $a = \{a_1(x), a_2(x)\}$，对于多层神经网络，我们做更普遍的规定，令第 l 层第 i 个节点的输入值为：

$$a_i^{(l)} = f(w_{i,1}^{(l-1)} \cdot a_1^{(l-1)} + w_{i,2}^{(l-1)} \cdot a_2^{(l-1)} + \cdots + w_{i,n}^{(l-1)} \cdot a_n^{(l-1)} + b_i^{l-1}) \tag{2-8}$$

将上式改写成用矩阵的表示方法，可以简化为

$$z^{(l)} = w^{(l-1)} \cdot a^{(l-1)} + b^{(l-1)} \quad (2\text{-}9)$$

$$a^{(l)} = f(z^{(l)}) \quad (2\text{-}10)$$

如果 MLP 中所有神经元均具有线性的激活函数，即该映射线性函数加权输入到每个神经元的输出上，通过线性代数的知识很容易证明出，任何数目的层均可化简为标准的两层输出模型，即标准的感知器单元）。因此多层感知器区别于单层感知器单元的地方在于，其中某些神经元使用了非线性激活函数，这些函数建模的原型包括大脑中的生物神经元电位模型，应激反应模型等。该激活函数存在的一般条件，即该函数需为连续函数。但理论上，激活函数可以是任何可微分函数。

MLP 至少包括三层感知层（输入层，输出层，一层或多层隐藏层），且包含具有非线性激活函数的节点。此结构被称作深度神经网络。一个 MLP 是一个层与层之间完全连接的网络，每一层的每个节点通过具有一定权重的边连接到下一层的每个节点。以此类推，输入经过神经网络这么多层的模拟，传播到输出端，因此也称为前馈神经网络。

2.1.3 反向传播算法

人工神经网络研究的另一个重要方面，是神经网络的学习规则。它要求迭代过程具有适当的学习速率，以便最小化实际输出和预测输出之间的误差。在诸多算法中，反向传播算法[26]被广泛用于神经网络的训练。反向传播算法基于梯度下降的原则，其思想在于将信号正向过程和预测误差与反向过程的学习阶段相结合，从而相应地更新权重矩阵。多层感知器网络通过反向传播算法完成训练，整个网络具有连接结构简单，学习状态稳定，易于在物理层面实现等优点。这种人工神经网络模型可以灵活运用于模式识别，图像分类，图片处理，函数拟合等方面。

下面我们将详细介绍反向传播算法的原理及推导过程。首先介绍根据反向传播算法计算多层复合函数的偏导数。假设复合函数

$$y = (x_1 + a)(x_2 + b)(x_1 + x_2) \quad (2\text{-}11)$$

根据常规的链式求导法则对输入求偏导，可以得到

$$\frac{\partial y}{\partial x_1} = \frac{\partial (x_1 + a)}{\partial x_1} \cdot \frac{\partial y}{\partial (x_1 + a)} + \frac{\partial (x_1 + x_2)}{\partial x_1} \cdot \frac{\partial y}{\partial (x_1 + x_2)} \quad (2\text{-}12)$$

$$\frac{\partial y}{\partial x_2} = \frac{\partial (x_2 + b)}{\partial x_2} \cdot \frac{\partial y}{\partial (x_2 + b)} + \frac{\partial (x_1 + x_2)}{\partial x_2} \cdot \frac{\partial y}{\partial (x_1 + x_2)} \quad (2\text{-}13)$$

在对输入进行求导时，y 对$(x_1 + x_2)$的求导被重复进行。如果 y 拓展为更庞大的多层复杂函数网络，那么这种重复运算的代价将是巨大的。

反向传播算法则避免了这个问题。它以输出节点为起始节点，反向求导，使

得每一条边仅被用于一次计算。以第 n 层与第 n - 1 层为例，输出 y 对第 n - 1 层第 m 节点的偏导数为 L，则 L =（第 n 层第 1 个节点对 m 节点的偏导数）*（y 对第 n 层第 1 个节点的偏导数）+（第 n 层第 2 个节点对 m 节点的偏导数）*（y 对第 n 层第 2 个节点的偏导数）+……+（第 n 层第 n 个节点对 m 节点的偏导数）*（y 对第 n 层第 n 个节点的偏导数）。以上式为例，从最上层的节点 y 开始，设初始值为1,以每一层为单位进行处理。y 的下一层包含的节点为(x_1+a),(x_1+x_2),$(b+x_2)$，分别将 y 对这三个节点内的值进行求偏导，所得的结果乘以 1 后存放在节点中，记为 *N1*，*N2*，*N3*。此时，倒数第二层传播完毕。倒数第三层包括四个节点x_1，x_2，a，b。a，b 为常数可忽略。将(x_1+a)对x_1求偏导数的值乘以 *N1*，记为 *M1* 存入x_1节点中；将(x_1+x_2)对x_1求偏导数的值乘以 *N2*，记为 *M2* 存入x_1节点中；将(x_2+b)对x_2求偏导数的值乘以 *N3*，记为 *M3* 存入x_2节点中；将(x_1+x_2)对x_2求偏导数的值乘以 *N2*，记为 *M4* 存入x_2节点中。执行 *M1+M2*，*M3+M4*，即可得到 y 对x_1和x_2的偏导数。这种方法下，*N2* 被重复利用，节约了计算资源。反向传播算法正是利用这一思路简化了求导过程。

利用反向传播算法训练时，将一组具有期望答案的数据输入到神经网络，初始时神经网络给出的输出预测会与真实期望有较大偏差，随着训练次数增多，每次使用优化算法（例如梯度下降法）不断修正边的权重 w 和偏置 b，使得网络的计算结果越来越接近标准答案，直至达到要求的精度。定义一个表达式 J，作为代价函数，它是关于 w 的函数。J 的具体形式可以有很多种，可根据不同情况定义，这里只需将它理解理解为计算误差的一种衡量。比较通用的代价函数为二次代价函数

$$J(W,b) = \frac{1}{2n}\sum_x (y - h(x))^2 \tag{2-14}$$

反向传播算法的目标就是使计算误差尽量小。以梯度下降法为代表的很多优化算法可以用来解决上述的最小化问题，通过对 J 求偏导数来对 w 与 b 进行微调

$$W_{ij}^{(l)}(t+1) = W_{ij}^{(l)}(t) - \alpha \frac{\partial}{\partial W_{ij}^{(l)}(t)} J(W,b) \tag{2-15}$$

$$W_{ij}^{(l)}(t+1) = W_{ij}^{(l)}(t) - \alpha \frac{\partial}{\partial W_{ij}^{(l)}(t)} J(W,b) \tag{2-16}$$

$$b_i^{(l)}(t+1) = b_i^{(l)}(t) - \alpha \frac{\partial}{\partial b_i^{(l)}(t)} J(W,b) \tag{2-17}$$

其中α表示学习速率，其大小通常介于 0 到 1 之间。我们引入残差δ来衡量预测值与期望值之间的距离。对于输出层 n，计算输出 h 与期望 y 的差值，δ即为输入对于输出的影响。

$$\delta_i^{(n)} = \frac{\partial J}{\partial z_i^{(n)}} \tag{2-18}$$

而对于隐藏层 l 来说，残差衡量了中间节点对最终偏差所负责的大小，它是输出对于每层节点输入的微分。

$$\delta_i^{(l+1)} = \frac{\partial J}{\partial z_i^{(l+1)}} = \frac{\partial J}{\partial a_i^{(l+1)}} \cdot f'(z_i^{(l+1)}) \tag{2-19}$$

$$\begin{aligned}
\frac{\partial J}{\partial a_i^{(l+1)}} &= \frac{\partial J}{\partial z_1^{(l+2)}} \cdot \frac{\partial z_1^{(l+2)}}{\partial a_i^{(l+1)}} + \frac{\partial J}{\partial z_2^{(l+2)}} \cdot \frac{\partial z_2^{(l+2)}}{\partial a_i^{(l+1)}} + \cdots + \frac{\partial J}{\partial z_n^{(l+2)}} \cdot \frac{\partial z_n^{(l+2)}}{\partial a_i^{(l+1)}} \\
&= \sum_{j=1}^{n} \frac{\partial J}{\partial z_j^{(l+2)}} \cdot \frac{\partial z_j^{(l+2)}}{\partial a_i^{(l+1)}} \\
&= \sum_{j=1}^{n} \delta_j^{l+2} \cdot \frac{\partial z_j^{(l+2)}}{\partial a_i^{(l+1)}} \\
&= \sum_{j=1}^{n} \delta_j^{l+2} \cdot W_{ji}^{l+1}
\end{aligned} \tag{2-20}$$

$$\therefore \delta_i^{(l+1)} = f'(z_i^{(l+1)}) \cdot \sum_{j=1}^{n} \delta_j^{l+2} \cdot W_{ji}^{l+1} \tag{2-21}$$

下面对 $\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b)$ 的展开进行简单的数学推导。由链式法则可得

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) = \frac{\partial J}{\partial a_i^{(l+1)}} \cdot \frac{\partial a_i^{(l+1)}}{\partial z_i^{(l+1)}} \cdot \frac{\partial z_i^{(l+1)}}{\partial W_{ij}^{(l)}} \tag{2-22}$$

分别计算每个子式的值，第一项可得上文中已求得：

$$\frac{\partial J}{\partial a_i^{(l+1)}} = \sum_{j=1}^{n} \delta_j^{l+2} \cdot W_{ji}^{l+1} \tag{2-23}$$

第二项可得：

$$\frac{\partial a_i^{(l+1)}}{\partial z_i^{(l+1)}} = f'(z_i^{(l+1)}) \tag{2-24}$$

第三项可得：

$$\frac{\partial z_i^{(l+1)}}{\partial W_{ij}^{(l)}} = a_j^{(l)} \tag{2-25}$$

三者相乘可得：

$$\frac{\partial}{\partial W_{ij}^{(l)}} J(W,b) = (\sum_{j=1}^{n} \delta_j^{l+2} \cdot W_{ji}^{l+1}) \cdot a_j^{(l)} \cdot f^{'}(z_i^{(l+1)})$$

$$= \delta_i^{(l+1)} \cdot (a_j^l)^T \quad (2\text{-}26)$$

对于 J 对 b 的偏导数可采用同样的方法求得：

$$\frac{\partial}{\partial b_i^{(l)}} J(W,b) = \delta_i^{(l+1)} \quad (2\text{-}27)$$

以上即为反向传播的推导过程。

反向传播这个术语常常被误解为在整体多层神经网络上施加的学习算法。实际上，反向传播算法仅用于在计算梯度时，而熟知的其他算法，例如随机梯度下降算法，则是使用该梯度执行学习训练。此外，反向传播算法也常常被误解为仅能用于多层神经网络训练，但原则上讲，它可以计算任何函数的导数。举一个具体例子来说，对于任意函数 f，我们需计算其梯度$\nabla_x f(x,y)$。其中 x 为一组变量，其导数的是我们通过反向传播算法预期求得的；而可将 y 视作函数输入的一组附加变量，其导数不是必需求得的。在学习算法中，我们最关注的是相对于参数的成本函数的梯度，$\nabla_\theta J(\theta)$。许多机器学习任务还会计算其他导数，用作学习过程的一部分或分析学习的模型。反向传播算法也可以应用于这些任务，并且不限于计算关于参数的成本函数的梯度。利用传播信息来计算导数的想法非常普遍，并且可以用于计算多输出函数的雅可比行列的值。

2.2 卷积神经网络基本思想

CNN 是一种前馈人工神经网络，网络中神经元之间的连接模式模拟动物视觉皮层的组织。单个神经元接受有限空间区域中的信息，称为感受野。不同神经元的感受野，使得它们平铺整个输入空间。单个神经元对其的刺激的响应可以通过卷积运算在数学上近似。CNN 在图像和视频识别，语音识别和自然语言处理中有广泛的应用。

相比于全连接神经网络，CNN 具有局部连接、权值共享和池化操作三大特性。这三大特性使得其在满足高效完成机器学习的前提下，提升了性能并优化了开销。本节详细介绍卷积神经网络的基本思想。

2.2.1 局部连接

反向传播神经网络中，层间节点连接采用全相连的模式，如图 2-3 所示。相比于反向传播网络，卷积网络通常具更稀疏的连接方式，这里称作局部连接[39]（Local Connection）。局部连接的结构如图 2-4。

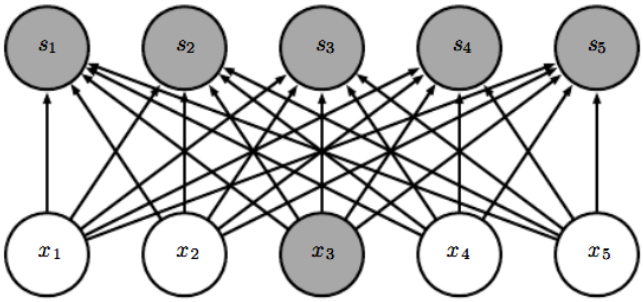


图 2-3 全连接方式

Fig.2-3 Full connection

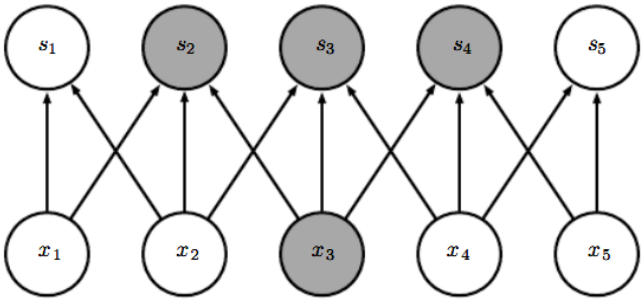


图 2-4 局部连接方式

Fig.2-4 Local connection

图 2-3 中是全连接的结构，神经元x_3与上层每一个神经元均相连；图 2-4 是局部相连结构，x_3仅与其上方的三个神经元相连。局部连接规定第 l 层的一个神经元最多只能连接到第 l+1 层的 k 个神经元上，k 远小于 l+1 层总神经元的数目 S。

局部连接能够极大减少训练参数。当处理图像时，输入图像可能有百万或千万级数量单位的像素，但我们可以仅检测那些细微的，有意义的特征部分，如边缘的信息，完成这项工作可能只会存储几百像素。例如，处理 1000×1000 像素的图像，假设采用的神经网络有 10^6 个隐藏层神经元，以全连接的方式，就有 10^{12} 个连接，也就是 10^{12} 个权值参数。如果图像的空间联系是局部的，一个神经元不需要对全局图像做感受，只感受局部的图像区域，即图 2-4 中，*S3* 不需要接受x_1和x_5的信息。这样，连接的数目即神经网络需要训练的权值参数的个数就大大减少。假如局部感受野是 10×10，则训练的参数仅有 10^8 个，比原来减少了四个数量级。

局部连接的意义在于，通过存储更少的参数，降低了模型的内存需求并提高其计算效率，也意味着计算输出需要较少的操作。这种改进对效率的提升效果相当大。如果有 m 个输入和 n 个输出，则矩阵乘法需要 $m\times n$ 个参数，实际使用的算法需要 O($m\times n$)的运行时间。如果我们限制每个输出连接数目为 k，则稀疏连接的方法仅需要 $k\times n$ 参数规模和 O($k\times n$)运行时间。对于许多实际应用，它可以

在保证机器学习任务完成的同时获得良好的性能。

2.2.2 权值共享

权值共享（Parameter Sharing）是指对模型中多个滤波器使用相同的参数。在传统的神经网络中，在计算本层的输出时，权重矩阵中的每个元素仅被使用一次。将它与输入向量中的一个元素相乘后，该值则再不会被访问到。权值共享的思想可以被解释为，网络具有相互绑定的权重，因为应用在某个输入上的权重值，同样会被用在其他位置。在卷积神经网络中，卷积核中的每个成员会被用在每个输入位置上（除了某些情况下的一些边界像素，取决于相关设计决定的边界）。图 2-5 描绘了权值共享的神经网络结构。

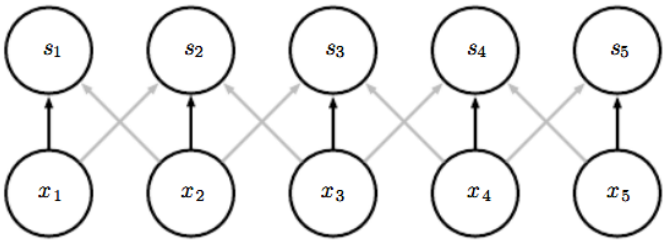


图 2-5 权值共享

Fig.2-5 Parameter Sharing

图 2-5 解释了权值共享对于参数利用率的提高效果。黑色箭头在权值共享结构中被重复用在每一个输入上；如果在全连接结构中，仅被一个节点使用一次，隐藏层的每一个神经元都连接 5 个图像区域，也就是说每一个神经元存在 5 个连接权值参数。假如这五个神经元用同一个卷积核去卷积图像，则五个神经元的参数是相同的。也就是说，网络只有 5 个参数。无论隐藏层有多少神经元，两层间的连接都只有 5 个参数。这就实现了权值共享。

同一个卷积核去卷积图像的做法只提取了一种特征。而图像需要提取多种特征，则可在神经元中加多种卷积核（滤波器）。假设我们使用 100 种滤波器，每种滤波器的参数不一样，表示它提取出的输入图像的不同特征，这样每种卷积核去卷积图像就得到对图像的不同特征的响应，称之为特征图（Feature Map）。这 100 个特征图就组成了一层神经元，即本层神经元共享这 100 个特征图。以上图为例，每一个特征具有 5 个参数，那么通过权值共享，这层的参数为 500 个。

卷积运算使用权值共享的思想是，我们不将每个位置的信息都作为独立的一组参数进行学习，简单的，我们仅学习一组参数。这不影响正向传播的运行的时间复杂度，它仍然保持在 O($k\times n$)，但它进一步减少了模型对 k 参数的存储需求。k 通常比 m 小几个数量级。因此卷积相比与需要密集计算的矩阵乘法来说，在存储器要求和统计效率方面有着更显著的优势。

2.2.3 池化操作

典型的卷积神经网络执行卷积工作时经历三个阶段。第一阶段，层并行执行几个卷积以产生一个线性激活的集合。在第二阶段中，集合中的元素经非线性激活函数变换。这个阶段称为检测阶段，输入数据在此阶段获得特征向量。在第三阶段为池化阶段，使用池化函数修改层的输出，如输出替换为某个位置附近输出的统计汇总。 池化操作实现的功能是将特征向量进行聚合，以减轻分类器的工作量，同时也防止了过拟合现象。

通过卷积之后，数据会获得自身的特征向量，这组特征向量的规模是十分庞大的。基于图像的知识我们知道，相邻像素之间的差异很小，可以将一个区域内的特征值进行平均或取最大值的操作，这样可以大大降低特征向量的规模。

对池化操作可以做下述分类：

1．一般池化（General Pooling）

1) 平均池化：对邻域内特征点求平均值，这种方式对背景保留更好，能减轻邻域大小受限造成的估计值方差增大的误差；

2) 最大池化：对邻域内特征点取最大值，这种方式对纹理提取更好，能缓解卷积层参数误差造成估计均值的偏移；

3) 随机池化：对像素点按照数值大小赋予概率，再按照概率选取，这种方式效果介于 1 和 2 之间；

2．重叠池化（Overlapping Pooling）

重叠池化操作中，池化的邻域之间会有重叠。

3．空间金字塔池化（Spatial Pyramid Pooling）

空间金字塔池化能够以任意尺寸的图片作为输入。

池化操作将整个邻域的响应进行汇总。它的一个单元会储藏 k 个像素的汇总数据，这使得池化单元的数目远小于检测单元的数目。由于下一层的输入减少了 k 倍，当下一层中的参数数量是其输入大小的函数（例如当下一层完全连接并基于矩阵乘法）时，池化操作也可以提高统计效率并减少存储参数的内存需求。

池化操作具有平移不变性。在选择连续区域做池化操作时，当输入平移（或旋转、伸缩）一个单元时，输出仍然会识别出输入的特征。如果我们更多关心是否存在某些特征而不是准确的位置时，这是一个非常有用的属性。

2.3 卷积神经网络结构

传统的 CNN 结构包含特征提取层和特征分类层。特征提取层由卷积层和池化层交替穿插出现，将特征整合输出。特征分类层由分类器构成，分类器将特征提取层的最后输出进行分类，完成学习过程。

2.3.1 整体架构

卷积神经网络结构整体是一系列表达层的组合。靠近输入端的隐层是由卷积层、激活层、池化层交替组成，不断从低维的图像中提取其抽象的高维特征，而靠近输出端则是由全连接层构成的 *Softmax* 分类器，这一部分即与传统的多层感知器一致。图 2-13 是经典的 LeNet-5 卷积神经网络整体架构[40]，包含 2 个卷积层，2 个全连接层，可用于手写识别。

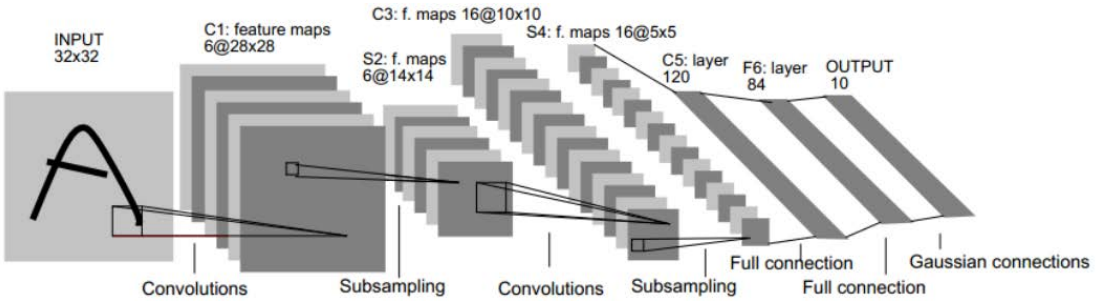


图 2-6 LeNet-5 网络结构[40]

Fig.2-6 LeNet-6 network architecture

2.3.2 卷积层

卷积层是 CNN 的核心组成部分，它具有局部连接和权值共享的特性。该层由带有学习能力的滤波器（卷积核）构成，它的感受野较小，但是可以延伸处理到输入数据的最大深度。在前向传播期间，每个过滤器都会完成具有输入宽度和深度规模的卷积运算，计算输入与过滤器输入的点积，并产生该滤波器的 2 维激活映射即特征图。通过这种网络学习方式，滤波器检测到输入在某些空间位置中包含的一些特定类型的特征时，就会产生对应的特征图。

将所有过滤器的特征图沿深度维度组合，即构成卷积层输出。输出的每个成员可以理解为一个神经元的输出，该神经元仅处理一个小的区域，并与其他神经元共享同一个特征图。

卷积层的输出规模由三个参数控制：深度，步长和填充值。深度决定了连接到输入图像同一区域的神经元数量，即神经元的个数，同时代表滤波器个数。所有这些神经元将学习输入图像的不同特征，并产生自己的特征图。例如，如果第一卷积层采用原始图像作为输入，则沿着深度维度的不同神经元会分别识别边的方向、团块、颜色等特征，并产生特征图。步长控制如何在空间维度（宽度和高度）分配深度列。例如当步长为 1 时，新的深度列的目标区域仅与原深度列目标区域有 1 个空间单位偏移。这较小的步长导致了感受野的重叠，造成输出规模巨大。相反，较高的步长，使得感受野重叠较少，输出规模较小。填充值即将输入

的边缘用零进行填充，方便从初始位置以步长为单位可以刚好滑倒末尾位置，通俗地讲就是为了总长能被步长整除。零填充对输出空间大小的进行控制。具体地讲，零填充将输入规模保持在了一个指定数值。

2.3.3 激活层

神经网络需要具有拟合非线性单元的性能，那就要把得到的结果输入一个非线性的激活函数即激活层，通过该函数映射后再输入池化层。

激活层也是神经网络结构中非常重要的组成部分，通过对卷积得到的结果进行适当的函数变换，模拟人脑神经元的激活过程，把适当的非线性因素加入到神经网络中，使其能处理数据线性不可分的情况，同时也能够把数据进行稀疏表达，使得处理更加高效。

之前应用中，经常使用的两种激活函数是 *Sigmoid* 和 *tanh* 函数：

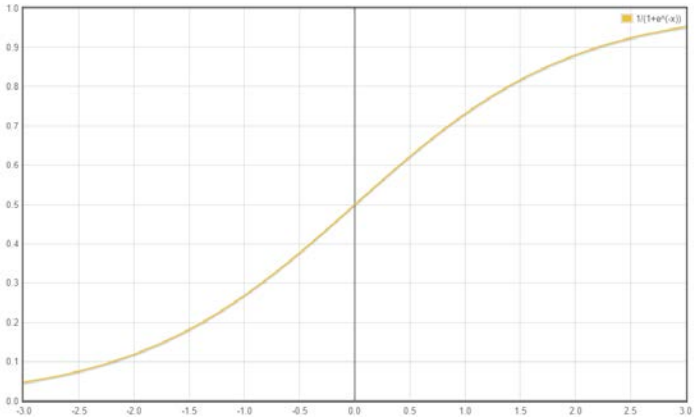


图 2-7 *Sigmoid* 激活函数

Fig.2-7 *Sigmoid* activation function

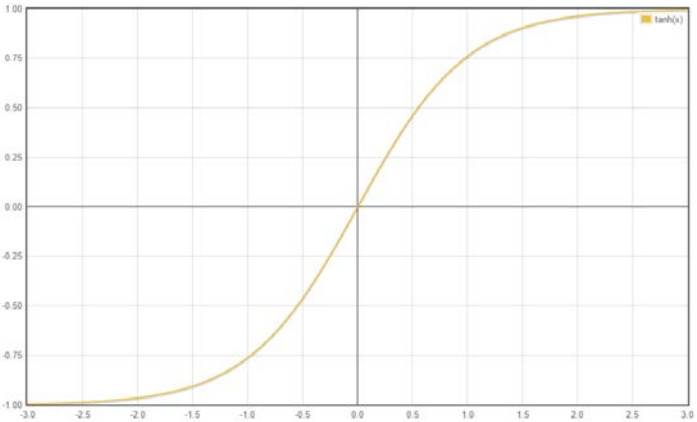


图 2-8 *tanh* 激活函数

Fig.2-8 *tanh* activation function

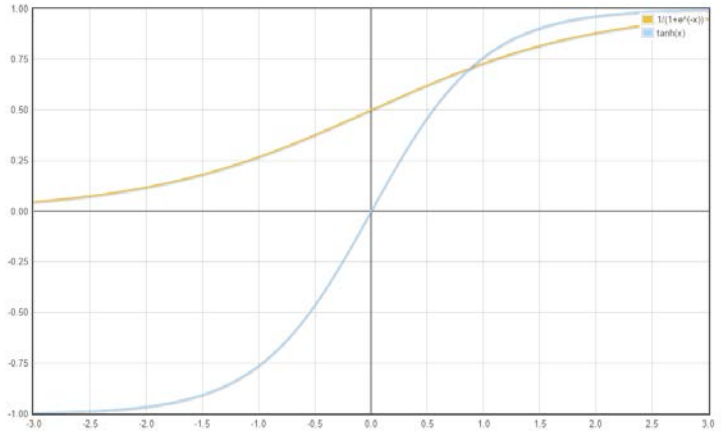


图 2-9 *Sigmoid* 和 *tanh* 区别

Fig2-9 Difference between *Sigmoid* and *tanh*

其中前者为一个 logistic 函数，值域为[0,1]。

$$f(x) = (1 + e^{-\lambda x})^{-1} \quad (2\text{-}28)$$

后者为一个双曲正切函数，形状上与前者相似，但值域为[-*a*, *a*]。

$$f(x) = atanh(bx) \quad (2\text{-}29)$$

其中，y 是一个节点（神经元）的输出，x 是突触输入值的加权和。

从数学上来看，非线性的 *Sigmoid* 函数对中央区的信号增益较大，对两侧区的信号增益小，在信号的特征空间映射上，有很好的效果。从神经科学上来看，中央区类似神经元的兴奋态，两侧区酷似神经元的抑制态，因而在神经网络学习方面，可以将重点特征推向中央区，将非重点特征推向两侧区。这两种方式都要比早期的线性激活函数要高明不少。

经过近些年来神经科学家的研究，模拟出了脑神经元接受信号更精确的激活模型，该模型如图 2-10 所示：

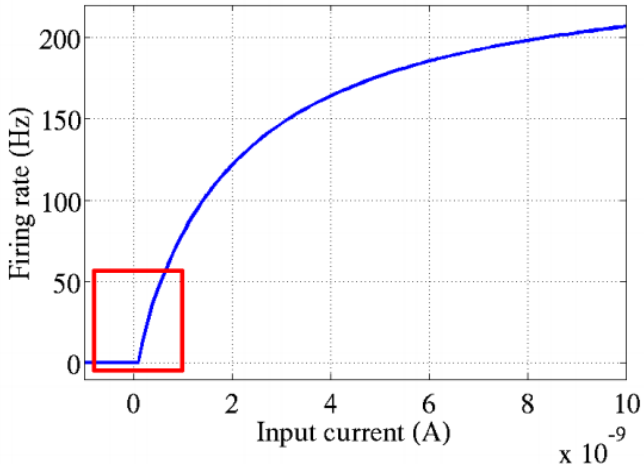


图 2-10 脑神经元激活模型

Fig.2-10 Neuron activation model

这个模型对比 *Sigmoid* 和 *tanh* 函数主要变化有三点：①单侧抑制 ②相对宽阔的兴奋边界 ③稀疏激活性（重点，可以看到红框里前端状态完全没有激活），因此 ReLU[41]逐渐替代之前的激活函数。ReLU 的表示为：

$$f(x) = \max(0, x) \tag{2-30}$$

其函数表示图为：

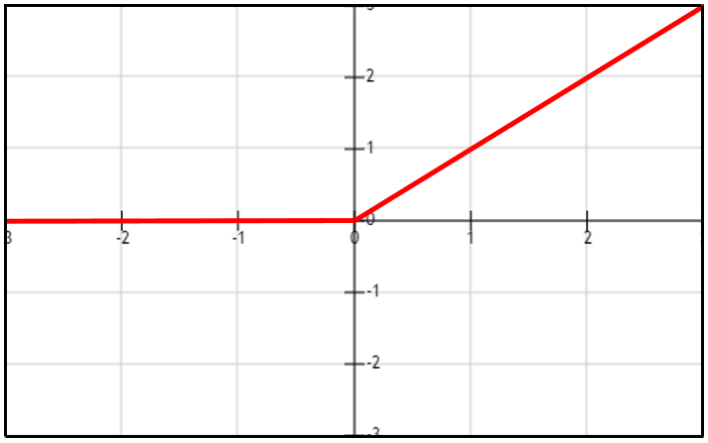


图 2-11 ReLU 函数

Fig.2-11 ReLU function

函数表示虽然简单，但是完全符合脑神经元接受信号的反应模型。从信号方面来看，即神经元同时只对输入信号的少部分选择性响应，大量信号被刻意的屏蔽了，这样可以提高学习的精度，更好更快地提取稀疏特征。从这个角度来看，在经验规则的初始化之后，传统的 *Sigmoid* 系函数同时近乎有一半的神经元被激活，这不符合神经科学的研究，而且会给深度网络训练带来巨大问题。从训练成本来看，由于神经网络训练过程中存在大量的求梯度的运算，使用 ReLU 激活函数要比 *Sigmoid* 系函数快的多。

在文献[18]中，作者对 ReLU 和普通 *Sigmoid* 系函数在训练时间方面做的对比测试，如图 2-12 所示。

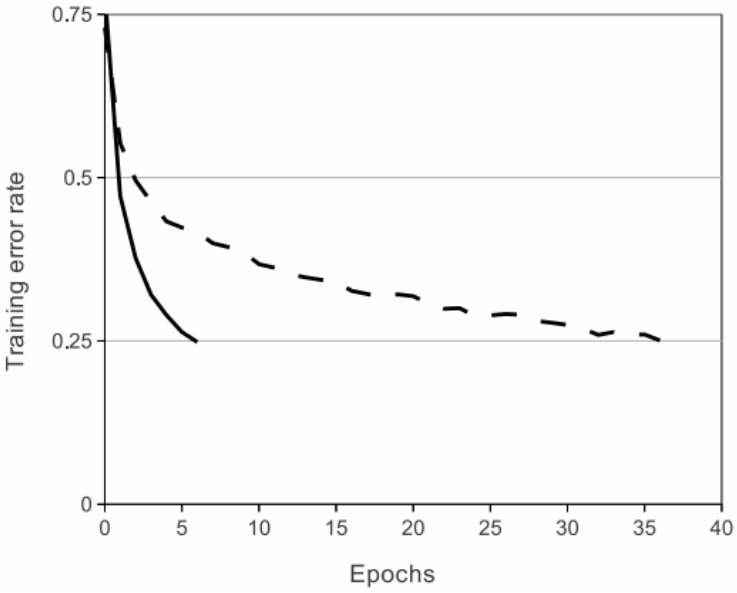


图 2-12 训练错误率和迭代次数关系[18]

Fig.2-12 Function between error rate and iteration times

图 2-12 是训练错误率与迭代次数的关系图。可以看出，在一个四层的卷积神经网络总，随着迭代次数的增加，训练错误率不断降低。而使用 ReLU 的模型达到 25%训练错误率时候所需要的迭代次数比使用 *tanh* 函数减少非常多。综合速率和效率，深度神经网络中大部分激活函数应该选择 ReLU。

2.3.4 池化层

池化层是采用池化操作对卷积层、激活层输出的特征图进行降采样。在常见的卷积神经网络架构中，卷积层与池化层交替穿插出现。

池化操作是一种非线性降采样。上一节已对池化操作进行了归类介绍，这里不再赘述。几个非线性函数中，最大池采样(max pooling)的应用最为广泛。它将输入图像划分成以组为单位的非重叠矩形区域，并且对每个这样的子区域进行聚合统计，输出子区域内的最大值。如图 2-12 所示，最大池采样子区域大小为 2×2，输入特征图为 4×4，将输入特征图划分为 2×2 的子区域，且子区域没有重叠，在每个子区域进行子采样后，对应输出特征图中的一个神经元。

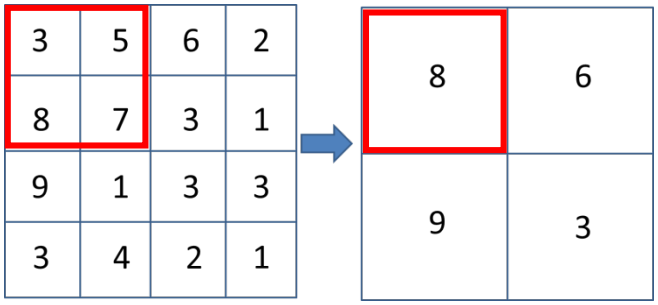


图 2-13 池化操作

Fig.2-13 Pooling operation

池化层的功能是减少输入规模大小，以减少网络中的参数的数量和计算负担，并由此防止过拟合。池化层通过进行池化操作保证了平移不变性，即使图像有在某个范围内小的位移，通过池化层提取的特征仍然保持不变。

2.3.5 分类器

在卷积神经网络中，感知学习功能由网络结构最后的分类器实现。分类器通过激活函数完成对输入的处理，通过梯度下降等算法，调整权值和偏置，逐步缩小感知预测结果和期望结果之间的距离，从而完成学习过程。大多数情况下，分类器采用 1 到 2 层全连接的人工神经网络。

理论上，分类器采用的激活函数可以是任意的对权值可微的函数。下文介绍的 *Softmax* 分类器是卷积神经网络中应用最为广泛的分类器。

Softmax 函数用于表示 *n* 个离散变量的概率分布。*Softmax* 函数常用作分类器的输出，来表示 *n* 个不同类别的概率分布。该分类器是逻辑斯蒂回归模型在多分类问题上的推广。

首先介绍逻辑斯蒂回归模型。逻辑斯蒂回归模型是为了解决二分类问题，因此 *y* 取值只有 0 和 1，其假设函数表达式为：

$$h_w(x) = \frac{1}{1+e^{-w^T x}} \tag{2-31}$$

其中，w是模型的参数，在训练过程中的代价函数为：

$$J(w) = -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)} log h_w(x^{(i)}) + (1 - y^{(i)}) log(1 - h_w(x^{(i)}))\right] \tag{2-32}$$

其中，$(x^{(i)}, y^{(i)})$，i =1,2,…,m，是训练样本集，m 是样本个数。我们的目标就是训练 w，使得代价函数最小。

在多分类问题中，分类标签 *y* 可以取两个以上的值。例如 *y* 可以取 *k* 个值，则对于输入 *x*，我们希望得到概率

$$y = P(Y = j \mid X) \tag{2-33}$$

也就是样本 X 被划分为某一类的概率。因此我们需要引入假设函数，该函数输出一个 k 维的向量，表示 x 属于该 k 个类别中的其中某个的概率，假设函数表达形式如式（2-34）：

$$h_{\omega}(x_i)=\begin{bmatrix} P(Y_i=1|x_i;\omega) \\ P(Y_i=2|x_i;\omega) \\ \cdots \\ P(Y_i=k|x_i;\omega) \end{bmatrix}=\frac{1}{\sum_{j=1}^{k} e^{\omega_j^T x_i}}\begin{bmatrix} e^{\omega_1^T x_i} \\ e^{\omega_2^T x_i} \\ \\ e^{\omega_k^T x_i} \end{bmatrix} \tag{2-34}$$

其中$\omega_1,\omega_2,\ldots,\omega_k$是模型的参数，$\frac{1}{\sum_{j-1}^{k} e^{\omega_i^T x_i}}$项是求和项，用作概率分布的归一化。

改写逻辑斯蒂回归模型的代价方程，可以得到：

$$\begin{aligned} J(\omega) &= -\frac{1}{m}\left[\sum_{i=1}^{m} y^{(i)}\log h_{\omega}(x^{(i)})+(1-y^{(i)})\log(1-h_{\omega}(x^{(i)}))\right] \\ &= -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=0}^{k} 1\{y^{(i)}=j\}\log p(y^{(i)}=j|x^{(i)};\omega)\right] \end{aligned} \tag{2-35}$$

利用梯度下降算法求解 J 的最小值，并且在迭代过程中对参数进行更新，从而更新全连接层的网络权值，从而得到最后的分类结果。

2.4 本章小结

本章内容主要分为三个方面。首先，介绍了人工神经网络的基础感知器单元和其组成的多层感知器，同时简要说明了神经网络的训练算法——反向传播算法；然后阐述了卷积神经网络相比与神经网络的特殊之处：局部连接、权值共享和池化操作。在此基础上，介绍了典型的卷积神经网络的组成部分：卷积层、激活层、池化层和分类器，以 Lenet-5 为例子简单阐释了 CNN 的整体框架，为后面章节的介绍打下基础。

第三章 深度卷积神经网络的算法研究与优化

3.1 卷积神经网络模型设计

本文选择人脸识别作为本文研究的应用场景，实现和优化 CNN 模型。本章主要从模型测试的准确率方面对网络模型进行设计和优化。

3.1.1 架构设计

CNN 模型包含四个卷积层（包含池化层），用来分层提取特征。然后是全连接层（这里称之为特征表示层）和指示身份类 *Softmax* 分类输出层，图 3-1 是深度 CNN 模型的详细结构，其输入采用 64×64 的 RGB 图，输出是预测当前输入图片属于哪个类别。

对于输入图像来说，输入大小为 64×64×k。k 代表输入图像是灰度图还是彩色图，如果 k = 3，那么输入为彩色图；如果 k = 1，那么输入为灰度图。在本次实验中，输出共有 10575 个类别。当输入大小更改时，网络层中特征图的高度和宽度也将相应更改。特征表示层的维度固定为 160 维，而预测输出层的维度根据其预测的类别数量而变化。特征数量随着特征提取层的增加而减少，直到形成高度紧密并且能够预测特征的最后一个隐藏层（特征表示层），通过少量的特征就能够预测相对较多的类别数。

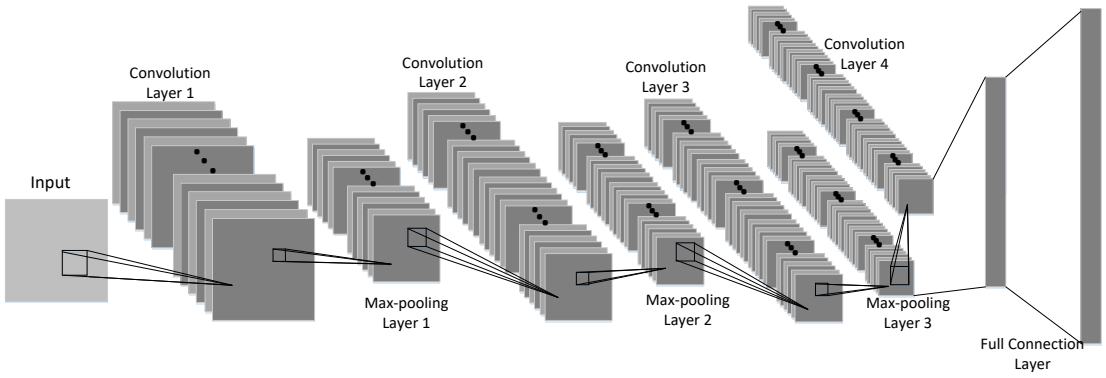


图 3-1 深度卷积神经网络架构

Fig.3-1 The architecture of deep convolutional neural network

在网络结构中，总共含有 4 层卷积层用以抽象提取每一层的特征。相比于浅层神经网络，4 层的卷积和 1 层全连接层相对已经比较深。对于计算机视觉领域的问题，任务越复杂，那么需要的网络层数就越多，需要训练的参数就越多。4 层卷积的网络模型能够初步解决人脸识别问题。卷积运算可以表示为：

$$y^{j(r)} = \max(0, b^{j(r)} + \sum_i k^{ij(r)} * x^{i(r)}) \quad (3\text{-}1)$$

其中，x^i和y^i分别代表第 i 幅输入图像和第 j 幅输出图像。k^{ij}表示第 i 幅输入图

和第 j 幅输出图之间的卷积核。*代表卷积操作。b^j是第 j 幅图像的偏置项。这里 CNN 模型使用 ReLU 处理隐层神经元。网络模型的较高卷积层中的权重在局部共享以学习不同区域中的中间或高级特征。公式（3-1）中的 r 代表权值共享的局部区域。在前三个卷积层中，权值在每一个 2×2 的区域中共享。

激活层采用 ReLU 激活函数，模拟人脑神经元的激活过程，在神经网络中加入一些适当的非线性因素，同时使得处理更加高效。

池化层采用最大池化策略，形式上表示为：

$$y_{j,k}^{i} = \max_{0 \le m,n < s} \{x_{j \cdot s+m, k \cdot s+n}^{i}\} \tag{3-2}$$

其中，在第 i 个输出图y^i中，每个神经元集中在第 i 个输入图x^i中的大小为 s×s 的非重叠局部区域。

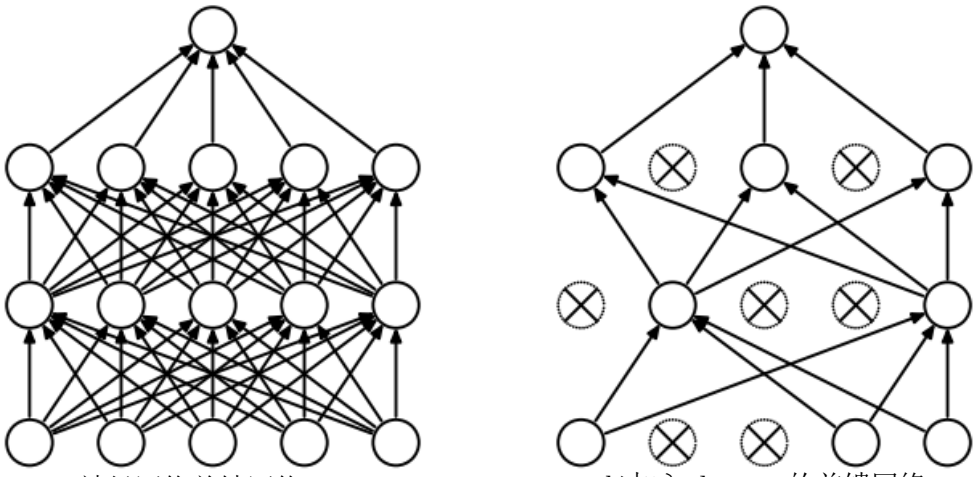
深度卷积神经网络的特征表示层是全连接层，在经过最大池化层之后，完全连接到第三个和第四个卷积层。因为第四卷积层是由第三卷积层经过池化和卷积操作得到，因此第四个卷积层中的特征更加抽象，第三和第四层的特征经过全连接能够使得最后的特征能够得到跟多尺度的特征[42]。这对特征学习至关重要，因为在级联方向上连续的降采样之后，第四卷积层包含比较少的神经元，并且可能会成为信息传播的瓶颈。在第三卷积层（称为跳过层）和最后一个卷积层之间增加旁路连接减少了第四卷积层中可能的信息损失。最后一个隐藏层接受函数

$$y_j = \max(0, \sum_i x_i^1 \cdot w_{i,j}^1 + \sum_i x_i^2 \cdot w_{i,j}^2 + b_j) \tag{3-3}$$

其中，x^1，w^1，x^2，w^2分别表示第三和第四卷积层中的神经元和权重。它线性地将前两个卷积层中的特征组合在一起，然后再将全连接的特征输入到 ReLU 激活层。

为了避免过拟合，全连接层经过 ReLU 激活函数后会经过 dropout 层[43]。过拟合是神经网络中比较严重的问题。dropout 是有效解决这个问题的方法。其关键思想是在训练期间前馈网络随机地丢弃神经元以及它们的连接。这能够防止一个神经元参与到的共同组合太多，更加符合人脑中稀疏响应的特性。

在神经网络训练过程中，隐层神经节点的激活输出值以一定的概率 p 输出 0，也就是说，这个神经节点与下一层网络某个节点断开连接，对其激活没有产生贡献。因此在反向传播算法更新网络权值时，不再更新与该节点相连的权值。dropout 在神经网络中的原理如图 3-2 所示。图 3-2 中图 a)表示的是正常神经网络的前馈过程，图 b)表示的是加入 dropout 的前馈过程，可以明显看出，在训练时候需要更新的权重减少，本层有些节点不参与下层节点的激活，这样的操作有助于避免过拟合。



a)神经网络前馈网络
a)Feedforward network

b)加入 dropout 的前馈网络
b)Feedforward network with dropout

图 3-2 dropout 前馈神经网络

Fig.3-2 Feedforward network with dropout

关于 dropout 为何能够提升神经网络的性能，下面从算法思想上进行直观的解释。

1. 在训练过程中，样本被输入神经网络进行权值更新时，隐层的节点是以比较小的概率被随机激活，所以隐层节点不一定会激活，所以在每次更新权值时也是随机的，不依赖于正常神经网络的固定关系，阻止了某些特征仅仅在其他特定特征下才有效果的情况。

2. dropout 也可以看作是模型平均的一种方式。由于 dropout 的存在，那么对于每次输入到网络中训练的图像，其对应的网络结构很可能都是不同的，但隐层节点更新后的权值又被所有的这些不同的网络结构共享。

深度卷积神经网络的输出是一个 *n* 路 *Softmax* 分类器，能够预测 *n* 个不同身份人的概率分布。

$$y_i = \frac{\exp(y_i')}{\sum_{j=1}^{n}\exp(y_j')} \tag{3-4}$$

其中，$y_j' = \sum_{i=1}^{160} x_i \cdot w_{i,j} + b_j$线性地组合最后特征层的 160 维特征向量。

3.1.2 网络参数

深度卷积神经网络模型每一层的参数如表 3-1 所示。

1) 数据输入层，输入采用 64×64 的 RGB 图。
2) 卷积层 conv1。该层含有 20 个 4×4 的卷积核，即对输入图像的每个 4×4 区域提取 20 种不同的特征送到下一层中。那么 conv1 含有需要训练的参数则为：每个卷积核含有 4×4=16 个参数，步长为 1 个像素，一共 20 个卷积核，输入图像大小为 64×64，那么总共需要训练的参数就是 320 个。

那么明显可以看出，CNN 的这种局部连接和权值共享的特性能够极大程度地减少参数的个数。

表 3-1 深度卷积神经网络模型配置
Table 3-1 Configuration of deep CNN model

类型	滤波器尺寸/步长（像素）	输出尺寸（像素）	参数数量（个）
conv1	4×4/ 1, 1	61×61×20	0.32k
pool1	2×2 / 2	30× 30 ×20	—
conv2	3×3 / 1, 1	28×28×40	7.2k
pool2	2×2 / 2	14×14×40	—
conv3	3×3 / 1, 1	12×12×60	21.6k
pool3	2×2 / 2	6×6×60	—
conv4	3×3 / 1, 1	4×4×80	43.2k
fc5	—	160	550.4k
fc6	—	—	—

3) 池化层 pool1。池化层采取的是最大池化操作，池化区域为 2×2，步长为 2，即无重复池化，选取区域内的最大值作为下一层对应点的特征点，经过池化层，特征图的长宽都会变为原来的一半。由于池化层只对卷积层做最大化操作，不涉及到对卷积层的线性或非线性变换，因此不需要训练参数。
4) 卷积层 conv2。该层还有 40 个 3×3 的卷积核，即对上一层特征图中的每个 3×3 区域提取 40 种不同的特征送到下一层中。那么 conv2 含有需要训练的参数则为：每个卷积核含有 3×3=9 个参数，步长为 1 个像素，一共 40 个卷积核，输入上一层的特征图为 20 幅，那么总共需要训练的参数就是 9×40×20 = 7200 = 7.2k 个。
5) 池化层 pool2。pool2 采用与 pool1 相同的方式对 conv2 卷积层输出的特征图进行操作。
6) 卷积层 conv3。该层含有 60 个 3×3 的卷积核，根据上文的分析，那么本层需要训练的参数是 9×40×60 = 21600 = 21.6k 个。
7) 池化层 pool3。pool3 最大池化层对 conv3 输出的特征图进行操作。
8) 卷积层 conv4。该层含有 80 个 3×3 的卷积核，根据上文的分析，那么本层需要训练的参数是 9×60×80 = 43200 = 43.2k 个。
9) 全连接层 fc5。全连接层是由 pool3 和 conv4 全连接而成的。由 pool3 全连接到 fc5 所需要训练的参数为：6×6×60×160 = 345600 = 345.6k 个。由

conv4 全连接到 fc5 所需要训练的参数为：4×4×80×160 = 204800 = 204.8k 个，总计 550.4k 个参数。

3.1.3 训练集

CASIA-WebFace[44]是中科院李子青组所收集创建的大规模人脸数据集，包含 10,575 个人的 494,414 幅图像。由于大数据和深卷积神经网络 CNN 的快速发展，人脸识别任务的准确率变得非常高，甚至可以与人类相比。几个研究团队使用各自的大规模训练数据集，在 LFW 数据集上达到了非常高的准确率。尽管有许多卷积神经网络的开源实现，但是大规模人脸数据集并没有公开可用的。然而，在人脸识别领域，当前的情况是数据甚至比算法更重要。为了解决没有公开数据集的问题，中科院李子青团队提出一种半自动的方式从互联网收集人脸图像，并建立这样一个大规模的数据集。数据集图片大小为 250×250。图 3-3 是从数据集中随机选取的不同人物的图像。由于是从互联网收集，这些数据在姿态、光照强度、遮挡以及分辨率上并不一致，即使是同一个人，也会有不同场合、姿态下的图像。因此，该数据集非常适合作为神经网络模型的训练集。

图 3-3 CASIA-WebFace 举例

Fig.3-3 CASIA-WebFace examples

3.1.4 训练流程

CNN 模型的训练主要分为前向和后向传播两个过程。算法流程如下：

(1) 初始化网络权值。在训练中本文使用标准差为 0.01 的高斯分布对网络进行初始化，偏置 b 设定为常数。

(2) 选取训练图像输入模型训练。训练样本的 batch 值是一次输入模型的图像数量，本文根据不同模型设定为 64/128。

(3) 前向传播。输入图像经过模型的各个表示层，得到对应的输出值。

(4) 反向传播。计算当前的输出值和预测值的差，判断是否满足截止条件，如果满足，则停止，否则利用 BP 算法，更新每一层网络的权重。

(5) 回到(2)迭代训练。

(6) 重新回到(2)，继续训练。

3.2 深度卷积神经网络模型

上一节设计了卷积神经网络模型用于人脸识别任务，结构功能清晰，复杂度比较小，但准确率并不是很高，无法达到理想的准确率。因此，针对如何提高人脸识别的准确率，本节主要研究通过增加卷积神经网络模型的深度即网络的层数来提高准确率。

3.2.1 架构设计

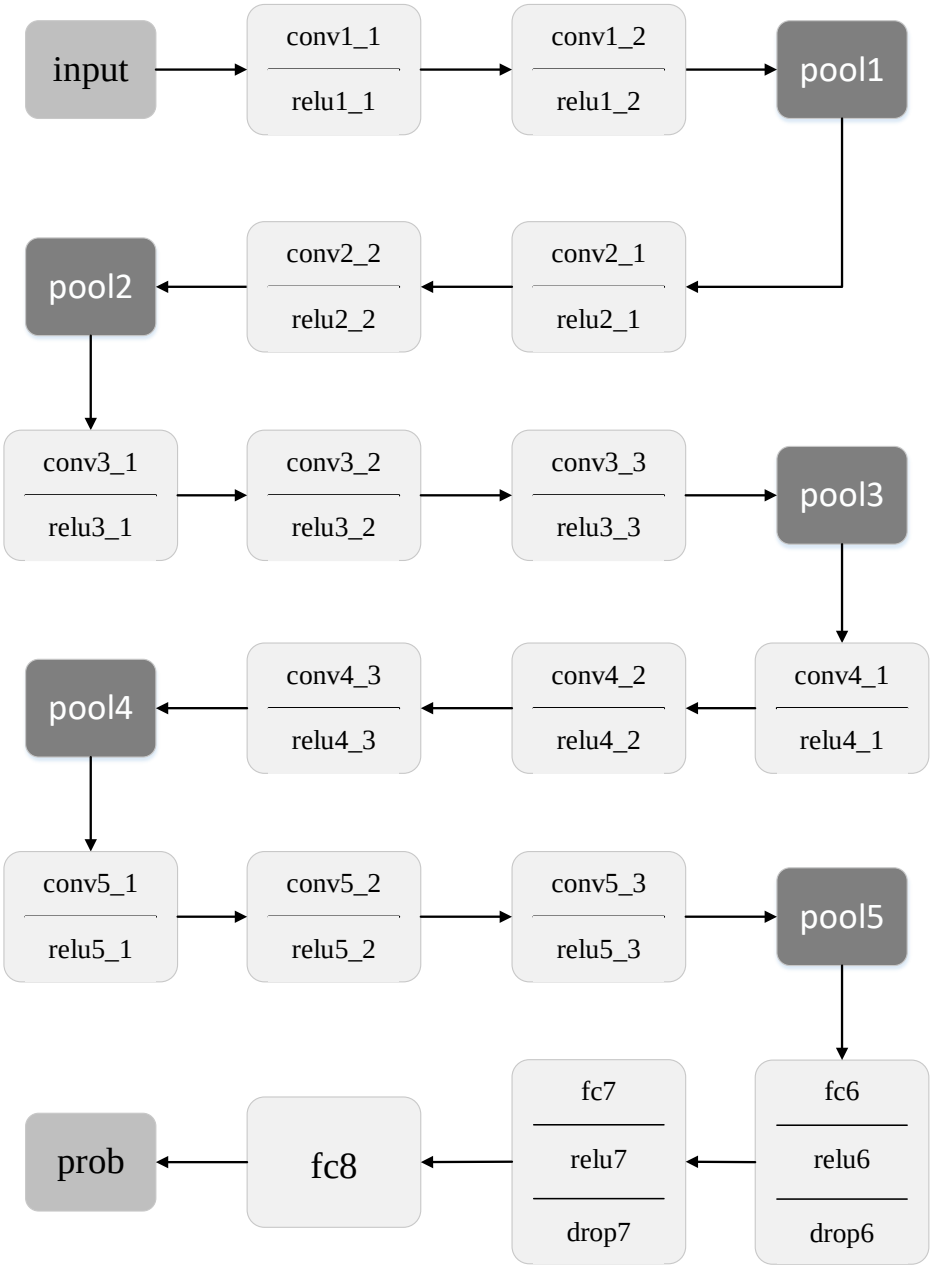


图 3-4 深度卷积神经网络模型

Fig.3-4 Improved deep CNN model

图 3-4 是改进的深度卷积神经网络结构。对于模型的输入，是固定大小的 224×224 的 3 通道 RGB 图像。在训练时候需要对图像进行预处理，在每个像素减去在训练集上计算的平均 RGB 值。

输入图像通过 13 个堆叠的卷积层以及 3 个全连接层，其中每个卷积层都使用具有非常小感受野的滤波器：3×3，这也是卷积滤波器的最小尺寸。在其中一个卷积层配置中，使用了 1×1 卷积滤波器，这可以看作是输入通道的线性变换，之后再接非线性变换。卷积层的步长设定为 1 个像素，卷积层输入的空间填充设

定为使得在卷积之后能够最大保持空间的分辨率，即，对于 3×3 的卷积滤波器，空间填充是 1 个像素。每个卷积层后都配置有 ReLU 来获得稀疏的特征表示。

整个网络模型的池化操作由五个最大池化层执行，会紧跟在每一层的卷积层之后。最大池化操作会在具有步长为 2 个像素的 2×2 窗口上执行。

在卷积层和池化层的堆栈之后是三个全连接层：前两个全连接层都具有 4096 个特征，第三个全连接层特征数量的多少可根据需要分类的类别数量来调整。最后一层是 *Softmax* 层。全连接层的配置在所有网络中都是相同的，和传统的神经网络一致。

3.2.2 网络参数

表 3-2 改进的深度卷积神经网络模型配置

Table 3-2 Configuration of improved deep CNN model

类型	滤波器尺寸/步长（像素）	输出尺寸（像素）	参数数量（个）
conv1_1	3×3 / 1, 1	224×224×64	0.576K
conv1_2	3×3 / 1, 1	224×224×64	36.8K
pool1	2×2 / 2	112×112×64	—
conv2_1	3×3 / 1, 1	112×112×128	73.7K
conv2_2	3×3 / 1, 1	112×112×128	147K
pool2	2×2 / 2	56×56×128	—
conv3_1	3×3 / 1, 1	56×56×256	294K
conv3_2	3×3 / 1, 1	56×56×256	589K
conv3_3	3×3 / 1, 1	56×56×256	589K
pool3	2×2 / 2	28×28×256	—
conv4_1	3×3 / 1, 1	28×28×512	1.18M
conv4_2	3×3 / 1, 1	28×28×512	2.36M
conv4_3	3×3 / 1, 1	28×28×512	2.36M
pool4	2×2 / 2	14×14×512	—
conv5_1	3×3 / 1, 1	14×14×512	2.36M
conv5_2	3×3 / 1, 1	14×14×512	2.36M
conv5_3	3×3 / 1, 1	14×14×512	2.36M
pool5	2×2 / 2	7×7×512	—
fc6	—	4096	102M
fc7	—	4096	16.8M
fc8	—	—	—

改进的深度卷积神经网络模型的每一层的配置在表 3-2 中，每行表示一层。从表中可以看出，卷积层的通道数目比较小，从第一层的 64 开始，在每个最大

池化层变为 2 倍，直到其通道数目达到 512。表 3-2 中最后一列列出了每一层所对应的参数的数量，参数数量的多少和模型的复杂度成正比。

在改进的网络结构中，所有的卷积层都使用的是 3×3 的感受野，而没有在低层的时候使用相对较大的感受野，比如 7×7，步长为 2 个像素的感受野。在卷积神经网络模型中，两个 3×3 的卷积层（中间没有最大池化层）堆叠在一起可以认为等效于一个具有 5×5 感受野的卷积层。那么，三个 3×3 的卷积层堆叠在等效于一个 7×7 感受野的卷积层。这里不直接用 7×7 而是用 3×3 的感受野，主要原因有两点：

第一，如果使用较大感受野的卷积层，那么在卷积层之后会有一个 ReLU 层加入非线性变换。如果使用较小的 3×3 感受野，即三个卷积层，每个卷积层都会跟一个 ReLU 层，那么结合三个 ReLU 层，这使得决策函数更具有辨别力。

第二，这样也可以减少参数的数量。假设三层 3×3 卷积堆栈的输入和输出都有 X 个通道，那么这三层卷积层就有$3 \times (3^2 \times C^2) = 27 \times C^2$个参数。如果直接使用单个 7×7 卷积层，那么将需要$7^2 \times C^2 = 49 \times C^2$个参数。相比于三层卷积层来说，参数大概增加了 81.5%。这可以看作是对 7×7 卷积层增加正则化，迫使它们通过 3×3 滤波器（在其间注入非线性变换）进行分解。

3.3 测试结果与分析

本文采用 LFW 数据集选取 6000 对图像测试人脸识别的准确率。从输入的一对图像提取一对特征向量，本文直接使用余弦相似度进行比对，判定这一对图片是否是同一个人。假设两个向量为 A 和 B，那么其余弦相似度为：

$$\text{similarity} = \cos\theta = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n}(A_i)^2} \times \sqrt{\sum_{i=1}^{n}(B_i)^2}} \quad (3\text{-}5)$$

模型的复杂度主要考虑模型大小和识别时间，即提取图像特征进行比对的时间。对本章的两个模型进行测试，结果如表 3-3 所示。

表 3-3 准确率复杂度对比

Table 3-3 Comparison of accuracy and complexity

网络模型	准确率	模型大小	识别时间
卷积网络模型	77.5%	19.7MB	0.0327s
深度卷积模型	92.2%	553MB	0.528s

从准确率来看，卷积网络模型受限于网络层数比较少和全连接层的维度只有 160 维，因此在现有相似度评价指标下，准确率较低。通过增加卷积层和全连接层，特征的维度有 4096 维，因此准确率明显升高。虽然准确率增加，但是也带来了模型复杂度的问题。从模型复杂度来看，后者的模型大小是前者的 28 倍，识别

时间也是前者的 16 倍。

3.4 本章小结

本章在第二章的基础上，设计了一个含有四层卷积层和一层全连接层的卷积神经网络架构，详细解释了每层网络中的操作以及在整个网络中发挥的作用。然后由此分析了架构中每一层网络的参数意义和所需要训练的参数数量，以及卷积神经网络的训练流程。虽然模型结构简单，功能明确，也能够取得不错的准确率，然而，对于人脸识别准确率要求较高的场合，此模型就不能满足要求。因此通过加深整个网络卷积层的深度来提取更多更高维的特征。在原有模型的基础上，增加了 9 个卷积层和 2 个全连接层，并且对网络参数也做了进一步的优化来确保能够提升人脸识别的准确率，但也带来了模型复杂度高等问题。

第四章 基于轻量卷积神经网络的算法研究与优化

针对第三章中深度卷积网络模型复杂度比较高的问题，通过分析其网络层次和参数数量，我们可以发现，复杂度的增加主要有两点原因。第一，准确率的提升伴随着卷积层数的增加，卷积层的增加会导致参数增多，从而模型的复杂度提升；第二，每经过一个嵌套的卷积层，特征图的数量就增加一倍。在深度网络模型中这会导致参数数量剧增。为了缓解这两个问题，本章主要研究一个轻量的 CNN 框架来降低模型复杂度。首先，为了在不减少卷积层数量的条件下减少模型参数，本文引入嵌套网络[45]（Network in Network，NIN），通过减少卷积层的卷积核大小来减少该层的参数数量；第二，我们从全连接层中引入 maxout 的概念到卷积层中，形成一个新的激活函数，称为最大特征映射[46]（Max Feature Map，MFM），特征图每经过一个 MFM 激活层，数量都会减少为原来的一半，这样能够避免出现深度网络模型中特征图数量过多的情况，同时可以获得更紧密的特征表示信息。

4.1 嵌套网络

传统的卷积神经网络一般来说是由：线性卷积层、池化层、全连接层堆叠起来的网络。卷积层通过线性滤波器进行线性卷积运算，然后再连接非线性激活函数，最终生成特征图。以 ReLU 激活函数为例，特征图的计算公式为：

$$f_{i,j,k} = \max(w_k^T x_{i,j}, 0) \tag{4-1}$$

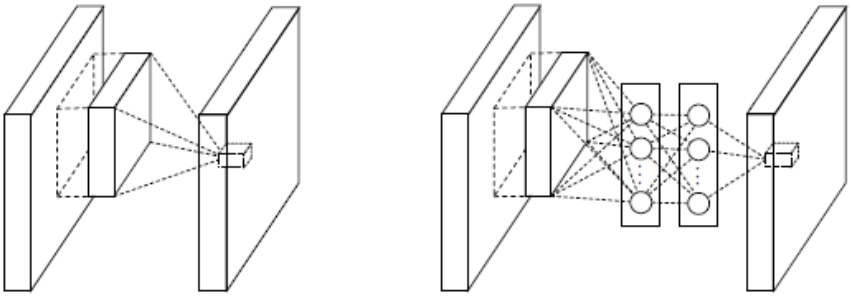
其中，(i, j)表示特征图中像素点的位置索引，$x_{i,j}$表示在(i, j)处卷积窗口中的图片块，k 则用来表示要提取的特征图通道的索引。

一般来说，如果要提取含有一些潜在特征的样本集是线性可分的话，那么对于线性的卷积运算来说这就足够了。然而，实现比较好的抽象过程的表示层通常是对输入数据的高度非线性表示，而一般来说所要提取的特征是高度非线性的。在传统的 CNN 中，这可以通过利用超完备的滤波器集合来补偿，用以提取各种潜在的特征。比如我们要提取某个特征，可以使用单独的线性滤波器去检测这个特征的不同形式，把所有可能的特征都提出来，这样就可以把想要提取的特征也覆盖到。然而，对于单个特征就是用这么多的滤波器会对下一层的网络造成非常大的负担，因为下一层网络需要考虑前一层特征所有变化的组合，这样会导致网络规模特别大，参数特别多。在 CNN 中，来自较高层的过滤器映射到原始输入中的较大区域。 CNN 高层特征其实是低层特征通过某种运算的组合。因此，可以认为，如果在把低层特征组合映射到高层特征之前，对低层特征进行更好的抽象，这样将会获得更好的特征。

4.1.1 多层感知卷积层

CNN 中的卷积滤波器是用于对基础数据块运算的广义线性模型，并且可以认为广义线性模型的抽象级别低。通过抽象过程提取出来的特征，意味该特征对于相同事物的变化是不变的。用更有效的非线性函数近似来代替广义线性模型可以增强局部模型的抽象能力。当含有潜在特征的样本是线性可分时，广义线性模型可以实现良好的抽象过程。因此，传统的 CNN 假设具有潜在特征的样本是线性可分的。

然而在实际中，这些样本数据常常是线性不可分的，所以需要对这些数据进行非线性运算提取高维的特征。在嵌套网络中，广义线性模型被替换为一种非线性的近似网络。同时这里我们选择多层感知器作为这种非线性网络的实例载体，即为多层感知卷积层。因为没有给定关于潜在特征分布的先验规律，那么更倾向于使用通用函数近似器用于局部块的特征提取，因为它能够对潜在特征的更抽象的表示进行近似表示。这里选择多层感知器有两个原因。首先，多层感知器与使用反向传播训练的卷积神经网络的结构兼容。第二，多层感知器可以是深层模型本身，这与特征重用的想法一致[47]。



a)线性卷积层　　b)多层感知卷积层

a)Linear convolutional layer　　b)Multilayer perceptron convolutional layer

图 4-1 线性卷积层和多层感知卷积层比较[46]

Fig.4-1 Comparison of linear conv layer and mlpconv layer

传统线性卷积层和多层感知卷积层的区别见图 4-1，线性卷积层和多层感知卷积层都将本地感受野映射到输出特征向量，而多层感知卷积层将输入的局部块映射到一个多层感知器的输出特征向量中，这个多层感知器由具有非线性激活函数的多个全连接层组成。通过与 CNN 类似的方式在输入上滑动多层感知器来获得特征图，然后将其馈送到下一层。由多层感知卷积层执行的计算如下：

$$f_{i,j,k_1}^{1} = \max({w_{k_1}^{1}}^{T} x_{i,j} + b_{k_1}, 0) \tag{4-2}$$

$$\vdots$$

$$f_{i,j,k_n}^{n} = \max({w_{k_n}^{n}}^{T} f_{i,j}^{n-1} + b_{k_n}, 0) \tag{4-3}$$

其中，*n* 是多层感知器的层数，ReLU 在网络中被用作激活函数。

从交叉通道（交叉特征映射）池的观点来看，式（4-3）等价于在正常卷积层上的级联交叉通道参数池。每个池化层对输入特征图执行加权线性重组，然后经过 ReLU 激活函数。交叉通道合并的特征图每进入下一层就要执行一次交叉通道合并。这种级联的交叉通道参数池结构允许在交叉信道信息上进行复杂的交互过程。交叉通道参数池化层也等效于具有 1×1 卷积核的卷积层。

4.1.2 嵌套网络结构

嵌套网络的整体结构是一个多层卷积感知层的堆叠，其顶层是全连接层和损失函数层。池化层建立在多层卷积感知层之间，与添加在正常的卷积神经网络中一样。图 4-2 显示了具有三个多层卷积感知层的嵌套网络。在每个多层卷积感知层内，存在三层感知器。嵌套网络和多层卷积感知层中的层数是灵活的，可以针对特定任务进行调整。

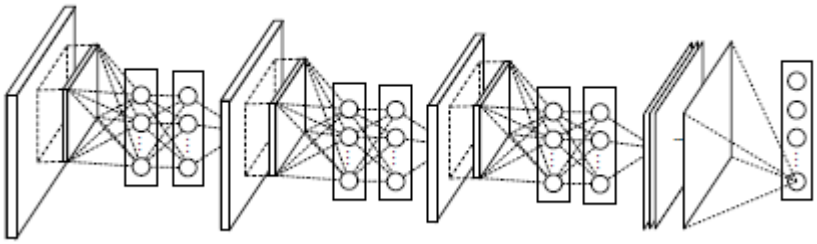


图 4-2 嵌套网络的结构[46]

Fig.4-2 Architecture of network in network

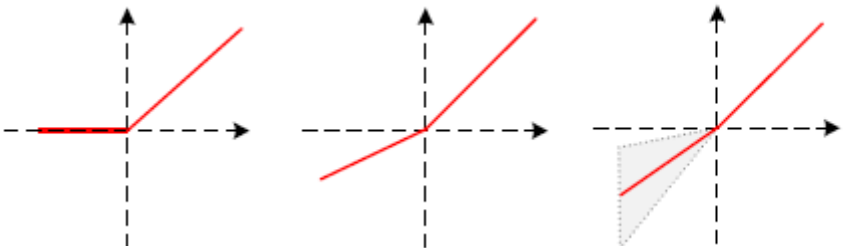
4.2 最大特征映射

由第二章可以看出，*Sigmoid* 或 *tanh* 激活函数是用于神经网络的非线性激活函数，经常会影响神经网络训练期间的鲁棒性优化[48]。当较低的网络层具有接近 0 的梯度时，它可能出现梯度消失的现象，由此影响网络的训练。因为较高层网络的梯度在-1 或 1 处几乎饱和。梯度消失现象可能导致卷积神经网络训练较慢或者收敛到一个比较差的局部最优。

ReLU 激活层通过提供网络单元的稀疏表示来克服梯度消失现象。尽管 ReLU 能够提升卷积神经网路的性能，但是 ReLU 在训练优化时候还是有可能的缺点。如果网络单元一直处于非活跃的状态，那么它的权值就是 0。这可能导致丢失一些信息，特别是对于前几个卷积层，因为这些层类似于 Gabor 滤波器，其正面和负面响应都应该考虑到。

为了解决这个问题，泄漏修正线性单元[49]（Leaky ReLU），参数修正线性单

元[50]（Parametric ReLU）和随机修正线性单元[51]（Randomized ReLU）被提出来缓解这个问题，如图 4-3。然而，这些激活函数也只是简单的分段线性激活函数，因此，在某些情况下它不能有效地表示特征。



a)修正线性单元 a)ReLU　b)泄露修正线性单元 b)Leaky ReLU　c)随机修正线性单元 c)Randomized ReLU

图 4-3 激活函数比较

Fig.4-3 Comparison of activation function

为了使前几层的特征表示更加紧密，而不是稀疏的 ReLU，我们这里使用最大特征映射激活函数。给定输入卷积层$C \in R^{h\times w\times 2n}$，如图 4-4 所示。

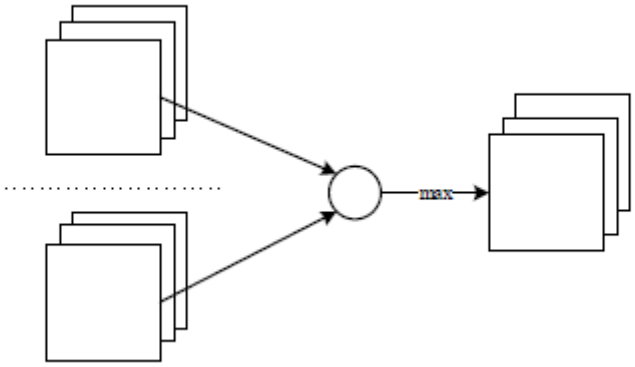


图 4-4 最大特征映射函数

Fig.4-4 Max feature map function

根据图 4-4，MFM 激活函数的表达式为：

$$f_{ij}^{k} = \max_{1\le k\le n}(C_{ij}^{k}, C_{ij}^{k+n}) \tag{4-4}$$

其中，$2n$ 为输入的卷积层通道数，$1 \le i \le h$，$1 \le j \le w$，函数的输出值域为 $R^{h\times w\times n}$。

根据式（4-4），这个激活函数的梯度函数如式（4-5）所示：

$$\frac{\partial f}{\partial c^{k'}} = \begin{cases} 1, \text{if } C_{ij}^{k} \ge C_{ij}^{k+n} \\ 0, \ otherwise \end{cases} \tag{4-5}$$

其中，$1 \le k' \le 2n$，

$$k = \begin{cases} k', & 1 \le k' \le n \\ k'-n, & n+1 \le k' \le 2n \end{cases}$$

从式（4-5）中可以看出，激活层 50%的梯度值都是 0。因此，MFM 激活函数像 ReLU 一样保留了稀疏梯度，同时这种稀疏梯度能够反映条件为响应变量的数据的方差。

MFM 激活函数不是像 ReLU 这样的单输入单输出函数，它是两个卷积特征图候选节点之间的最大值。这种激活函数利用聚合统计的方法，不仅可以获得特征的相对紧密的表示，而且可以获得实现变量选择和尺度降低的稀疏梯度。此外，最大特征映射函数还可以被看作两个卷积层之间的稀疏连接，其将信息稀疏地编码成特征空间。

综上所述，最大特征映射函数可以用来替代 *Sigmoid* 或者 ReLU 作为激活函数，来获得更紧密的特征表示和特征选择。带有最大特征映射层的轻量卷积神经网络的架构如图 4-5。

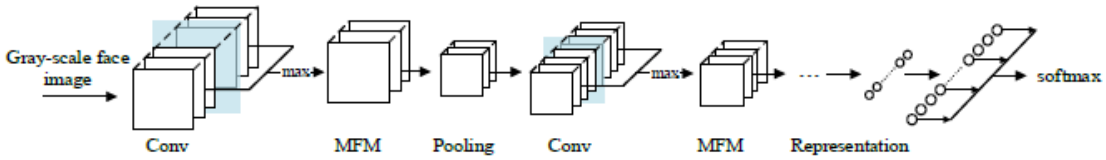


图 4-5 轻量卷积网络模型架构[45]

Fig.4-5 Architecture of lightened CNN

4.3 轻量卷积神经网络模型设计

4.3.1 架构设计

图 4-6 是轻量卷积神经网络的模型结构。从模型结构可以看出，假设 n 个特征图通过卷积层进入 MFM 激活层，通过 MFM 操作，特征图的数量会减少为 $n/2$。每经过 MFM 层就会减少一半的参数；与此同时，通过引入嵌套网络，使用尺寸为 1×1 的卷积核对特征图进行卷积操作，即每个卷积滤波器的参数会降低为原来的 $1/n$。在保证网络层数不变的情况，能够最大限度地减少网络参数，因此称之为轻量卷积神经网络。

在网络结构中，总共含有 9 个卷积层（其中有 4 个嵌套网络）、5 个池化层和 2 个全连接层。模型的输入是 128×128 的灰度图，在训练时候需要对图像进行预处理，并且 resize 到模型要求的输入。训练集使用 CASIA_WebFace 数据集，因此训练时 *Softmax* 分类器输出 10575 个类。

在网络结构中，总用含有 4 个卷积核为 1×1 的卷积层，即之前介绍到的嵌套网络。通过这样的卷积核在不影响其他卷积层的感受野的情况下能够增加决策函数的非线性程度。即使在本网络结构中，1×1 卷积实质上是对相同维度的空

间的线性投影，可以理解为特征图在做卷积的过程中形成交叉通道参数池，也能够在一定程度上增加模型的泛化能力。

激活层根据不同层分别采用 MFM 函数或者 ReLU 激活函数。在前三个卷积层中，使用最大特征映射函数，能够得到比较紧密的特征信息，相比于 ReLU 激活函数来讲，如果网络单元处于非活跃的状态，那么它的权值就是 0。这可能导致丢失一些重要的信息，特别是对于前几个卷积层。在后面的卷积层中主要使用 ReLU，一方面是为了给网络模型增加一些稀疏性，另一方面是为了提升网络的训练速度。

经过若干个卷积层和池化层之后，产生 512 维的一维全连接层来表示特征。之后经过 ReLU 激活层和 dropout 层来避免过拟合。

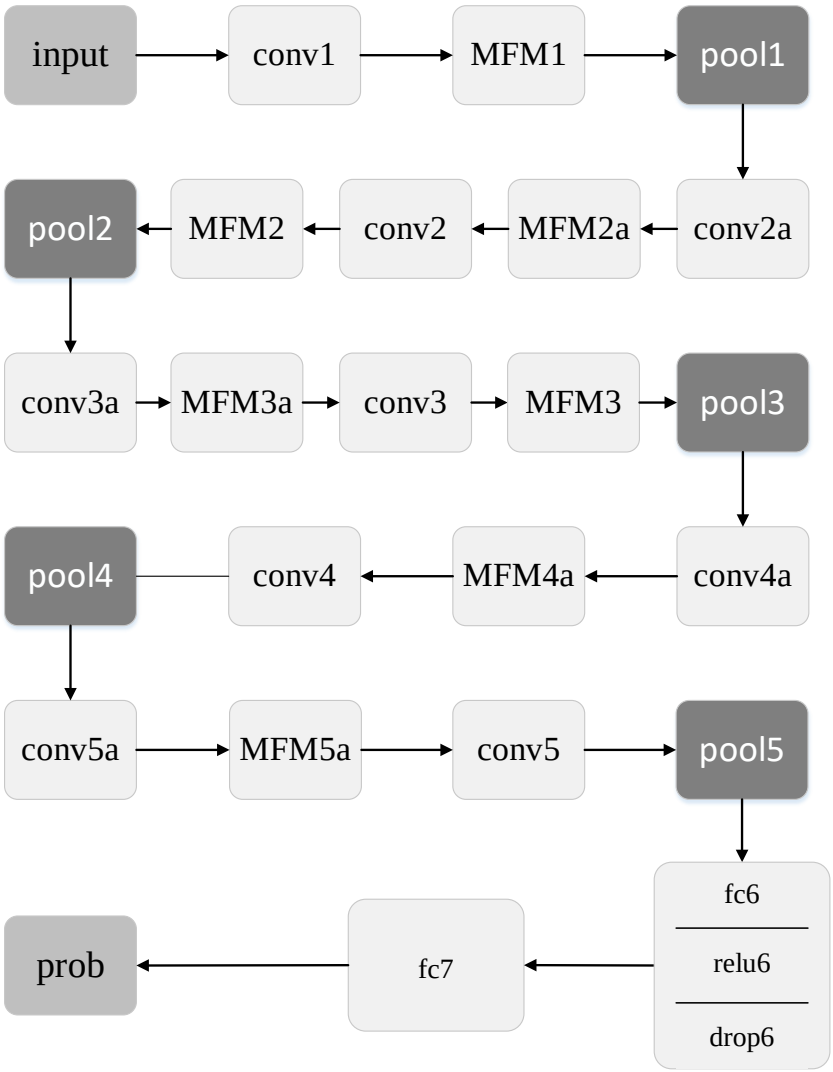


图 4-6 轻量卷积神经网络模型

Fig.4-6 Lightened CNN model

4.3.2 网络参数

轻量卷积神经网络的模型的每一层的配置如表 4-1 中所示。每一层网络参数计算参考 3.2.2 小节，具体数量如表 4-1 中最右边一列。

表 4-1 轻量卷积神经网络模型配置
Table 4-1 Configuration of lightened CNN model

类型	滤波器尺寸/步长（像素）	输出尺寸（像素）	参数数量（个）
conv1_1	5×5 / 1, 2	128×128×48	1.2k
conv1_2	5×5 / 1, 2	128×128×48	1.2k
MFM1	—	128×128×48	—
pool1	2×2 / 2	64×64×48	--
conv2a	1×1 / 1, 1	64×64×96	4.6k
MFM2a	—	64×64×48	—
conv2_1	3×3 / 1, 1	64×64×96	41.5k
conv2_2	3×3 / 1, 1	64×64×96	41.5k
MFM2	—	64×64×96	—
pool2	2×2 / 2	32×32×96	—
conv3a	1×1 / 1, 1	32×32×192	18.4k
MFM3a	—	32×32×96	—
conv3_1	3×3 / 1, 1	32×32×192	165k
conv3_2	3×3 / 1, 1	32×32×192	165k
MFM3	—	32×32×192	—
pool3	2×2 / 2	16×16×192	—
conv4_a	1×1 / 1, 1	16×16×384	73.7k
MFM4a	—	16×16×192	—
conv4	3×3 / 1, 1	16×16×256	442k
pool4	2×2 / 2	8×8×256	—
conv5_a	1×1 / 1, 1	8×8×256	65.5k
MFM5a	—	8×8×128	—
conv5	3×3 / 1, 1	8×8×256	295k
pool5	2×2 / 2	4×4×256	—
fc6	—	512	2.10M
fc7	—	—	—

对比第三章中的两个网络模型可以看出，相比于 CNN 模型，明显轻量卷积神经网络模型要更深一些，基本与深度卷积神经模型的深度相同。不过从参数数量来看，本模型介于前两者之间。基础模型的每一层特征图的数量累加 20，因此提取的特征比较少;且由于特征表示层只有 160 维向量,所以参数也相对少一些。而改进模型提取的参数明显多很多，不仅特征图数量增加，与此同时，2 层全连

接层都有 4096 维向量，因此需要训练的参数非常多，达到了 133M 之多。而在本模型中，由于最大特征映射的存在，每次经过最大特征映射激活层后，特征图的数量会减半，同时全连接层只有 512 维。因此本模型参数数量为 3.40M 左右。

4.4 测试结果与分析

根据第三章的测试方法对本章设计的轻量卷积网络模型进行准确率和复杂度的测试，并与第三章深度卷积模型进行对比，如表 4-2 所示。

从复杂度来看，轻量卷积网络的模型大小只有深度卷积模型大小的 1/17，同时识别时间是深度卷积模型的 1/5。可以得出，通过引入嵌套网络和最大特征映射激活函数，能够有效降低减少模型的参数，减低模型的复杂度。从准确率来看，轻量卷积模型相比于深度模型还有一些差距。一方面是由于全连接层维度减少为 512 维，另一方面是由于余弦相似度的评价方法并不能全面展示特征的分类能力。

表 4-2 准确率复杂度对比

Table 3-3 Comparison of accuracy and complexity

网络模型	准确率	模型大小	识别时间
深度卷积模型	92.2%	553MB	0.528s
轻量卷积模型	81.9%	31.2MB	0.103s

4.5 本章小结

在第三章的基础上，深度卷积神经网络虽然正确率有所提升，但是模型的复杂度也成倍增长，因此本章主要考虑在不减少准确率或者尽量少降低准确率的情况下，降低模型的复杂度。嵌套网络和最大特征映射是其中优化卷积神经网络的两种办法。嵌套网络通过加入一层 1×1 的卷积层，对进入卷积层的特征进行一次抽象，使得提取到的特征更具有代表性。最大特征映射激活层能够降低一半的特征图，因此也能够通过控制特征图的数量来降低模型复杂度。同时，全连接层降至一层，网络参数也进行相应的优化来减少参数数量。通过理论分析，轻量卷积神经网络能够极大程度降低模型复杂度，同时较深的网络也能保证具有一个不错的人脸识别准确率。

第五章 实验验证与结果分析

本章我们主要给出利用深度学习框架对前文提出的三种 CNN 模型进行测试验证的结果。首先介绍了本文搭建 CNN 模型所使用的 Caffe 深度学习框架和测试数据集 LFW 人脸库，然后给出了数据的预处理方式，包括人脸检测、对齐和图像剪切。随后给出了详细的准确率和复杂度的测试方法，对第三章和第四章的三种模型的准确率和复杂度测试结果进行比对分析，最后给出了轻量 CNN 模型与近期比较经典的 CNN 模型的复杂度对比分析。

5.1 Caffe 深度学习框架概述

Caffe[52]，全称 Convolutional Architecture for Fast Feature Embedding，是一个计算卷积神经网络相关算法的框架。Caffe 是一个清晰，可读性高，快速的深度学习框架。Caffe 是由贾杨清博士创建，由加州大学伯克利分校的视觉与学习中心和社区贡献者共同开发的。

5.1.1 基本特性

Caffe 提供了一个完整的工具包，用于培训，测试，调试和部署模型，同时也提供了所有这些任务和功能的详细文档示例。因此，对于希望进入机器学习的研究人员和开发人员来说，这是一个理想的起点。同时，它可能是这些算法中最快的可用实现，使其立即可用于工业开发。Caffe 框架有以下几个特点：

在模块集成方面，该软件从一开始设计就尽可能模块化，允许扩展新的数据结构，网络层和损失函数。许多层包括卷积层，池化层和损失函数已经实现，并且大量的示例展示了如何将它们组合成可用于各种任务的可训练的识别系统。

在特征表示和实现方面，Caffe 模型定义使用协议缓冲语言为配置文件。Caffe 以任意有向无环图的形式支持网络体系结构。在实例化时，Caffe 保留网络所需要的内存，并从其在主机或 GPU 中的底层位置抽象。通过内部的一个设置即可在 CPU 和 GPU 实现切换。

在测试方面，Caffe 中的每个模块都会进行一次测试，只有进行相应的测试才会把新的代码提交到项目中。这允许快速改进和重构代码库，并给使用代码的研究人员带来了比较好的用户体验。

在快速开发方面，对于快速开发原型和与现有代码连接，Caffe 提供了 Python 和 MATLAB 绑定。这两种语言都可用于构建网络和对输入进行分类。Python 绑定还公开了解决方案模块，以便轻松地创建新的训练过程。

在模型参考方面，Caffe 为视觉任务提供了参考模型，包括 R-CNN 模型、

AlexNet 模型等等。作者希望通用的软件底层能促进在网络架构和应用程序搜索的快速进展。

5.1.2 数据结构

Caffe 主要包含 Blob、Layer、Net、Solver 这几个大类。这四个大类自下而上，环环相扣，贯穿了整个 Caffe 的结构，Blob 是基础的数据结构，是用来保存学习到的参数以及网络传输过程中产生数据的类。Layer 是网络的基本单元，由此派生出了各种层类。修改这部分的人主要是研究特征表达方向的。Net 是网络的搭建，将 Layer 所派生出层类组合成网络。Solver 是 Net 的求解，修改这部分可以研究深度学习的求解方向。

作为数据传输的媒介，无论是网络权重参数，还是输入数据，都是转化为 Blob 数据结构来存储。直观来说，可以把 Blob 看成一个有 4 维的结构体（包含数据和梯度）。Blob 提供统一的存储器接口，保持一批图像（或其他数据），参数或参数更新。Blob 通过根据需要从 CPU 主机同步到 GPU 设备来隐藏混合 CPU / GPU 操作的计算和心理开销。在实际操作中，将数据从磁盘加载到 CPU 代码中的 Blob，调用 CUDA 内核以进行 GPU 计算，并将块转移到下一层，这样能够忽略底层细节，同时保持高水平的性能。主机和设备上的内存按需分配（延迟），以实现高效的内存使用。

模型以 Google 协议缓冲形式保存到磁盘，它有几个重要的特点：当串行化时能够最小化二进制字符串，高效序列化，可读性很强的文本格式，以及多种语言的高效接口实现。数据都存储在 LevelDB 数据库中。在我们的测试程序中，LevelDB 和协议缓冲形式在商用机器上提供了 150MB/s 的吞吐量，且对 CPU 的影响最小。

作为网络的基础单元，Caffe Layer 是神经网络层的本质，神经网络中层与层间的数据节点、前后传递都在该数据结构中被实现。它需要一个或多个 Blob 作为输入，并产生一个或多个 Blob 作为输出。Layer 对于网络作为整体的操作具有两个关键作用：接收输入并产生输出的前向通道，以及相对于输出梯度后向传播的通道。Caffe 提供了一套完整的层类型，包括：卷积，池化，内积，非线性，如整流线性和逻辑，局部响应归一化，元素操作和损失函数。这些都是完成计算机视觉任务所需要的类型。

作为网络的整体骨架，决定了网络中的层次数目以及各个层的类别等信息。Caffe 会标记所有的有向无环图层，确保前向和后向传播的正确性。Caffe 模型是端到端的机器学习系统。典型的网络从数据层开始，数据层从磁盘加载，并以损失层结束，损失层计算任务的目标，例如分类或重建。

作为网络的求解策略，涉及到求解优化问题的策略选择以及参数确定方面。

这个类中包含一个 Net 的指针，主要是实现了训练模型参数所采用的优化算法，它所派生的类就可以对整个网络进行训练了。

5.2 测试数据处理

5.2.1 测试集

本文使用 LFW[2]数据集作为测试集。LFW 数据集提供一组标记的人脸照片，这些面部照片包含人们日常生活中通常遇到的条件范围。数据集图片在姿势、照明、焦点、分辨率、面部表情、年龄、性别、种族、配件、化妆、遮挡、背景和照片质量方面都非常自然。数据集包含 5749 个人的 13233 张的彩色图片，其中的 1680 人拥有两张及以上的图片。一些图像可能包含多个脸部，但只有包含图像的中心像素的脸部，其被认为是图像的定义脸部。除目标脸部以外的都会被忽略为背景。数据集图片大小为 250×250，大部分是彩色图，一小部分是灰度图。图 5-1 是从数据集中随机选出的人物的图像。

在实验结果测试中，LFW 数据集被用作测试集，用来测试数据集中给定的两张图片是否是同一个人。在数据集中随机选出 *n* 对这样的图片，通过训练好的卷积神经网络模型来判断有多少对是判断正确的，多少对是判断错误的。这样可以大致估算模型的准确率。图 5-2 是显示的是数据集中部分匹配和不匹配的图像。

图 5-1 LFW 数据集概览

Fig.5-1 LFW dataset overview

图 5-2 匹配和不匹配的图像对比

Fig.5-2 Comparison of matched and mismatched images

5.2.2 预处理

许多实践中的应用程序希望自动检测、对准和识别较大静止图像中的人脸区域或较大场景的视频中的人脸区域。因此，人脸识别通常自然地被描述为检测-对齐-剪切-识别流程的一部分，如图 5-3 所示。

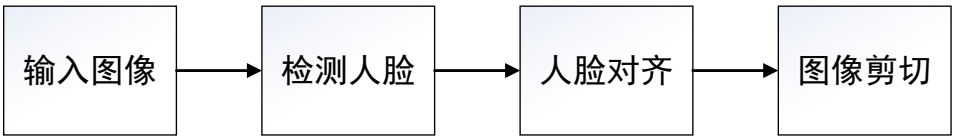


图 5-3 预处理流程

Fig.5-3 Process of pre-processing

因此，在选择图像进行训练时，需要对图片进行检测-对齐-剪切操作，以保证卷积神经网络模型只对兴趣区域产生响应。同样地，在测试时候，也需要对测试图像对进行检测和对齐，最后再进行识别的过程。

1. 人脸检测

首先利用香港中文大学公开的人脸检测器 FaceDetect.exe 对图像进行人脸检测，可以检测出训练集中哪些图像未含有人脸，之后进行人脸对齐和训练时可以把这些图像剔除出去。以 CASIA_WebFace 中李小龙的图像为例，检测到的人脸图像如图 5-4 所示。

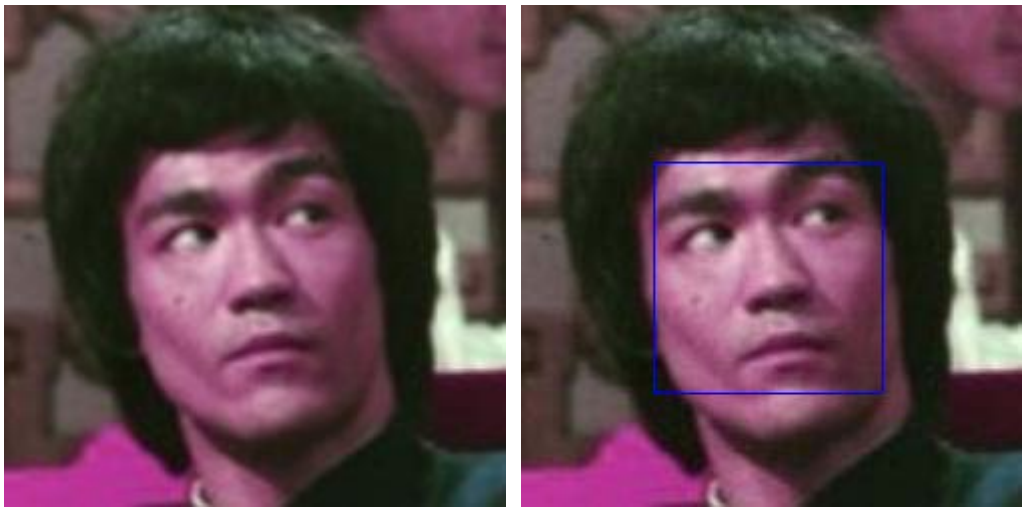


图 5-4 人脸检测
Fig.5-4 Face detection

图中蓝色画框的是检测到的人脸。在数据集中，也有一些图像无法检测到人脸，如图 5-5 所示。

图 5-5 检测失败图像
Fig.5-5 Detection failed image

2. 人脸对齐

在检测到人脸之后，通过人脸对齐，能够有效提取人脸特征，进而相同的人脸通过训练好的特征提取器进行比对，可以获得更加可靠的结果。

人脸可以比较容易地通过面部的特征点来对齐。在检测到的人脸上进行特征

点检测，获得人脸上一系列明显的特征点。预先定义一组模板，代表当前人脸是在正常位置上。利用待对齐图像检测到的特征点和模板的特征点进行比对，计算其仿射矩阵 H；然后再利用这个仿射矩阵 H，可以直接计算得到待对齐图像对齐后的人脸图像。

那么，首先需要检测到人脸的特征点。本文采用香港中文大学公开的人脸特征点检测器 FacePointDetect.exe 对图像中人脸的特征点进行检测。还是以 CASIA_WebFace 中李小龙的图像为例，检测到的人脸图像如图 5-5 所示。本文中，默认选取的参考图像是 CASIA_WebFace 数据集的第一张图像，即图 5-4 的左图。图 5-6 展示了经过人脸检测和特征点检测的图像。

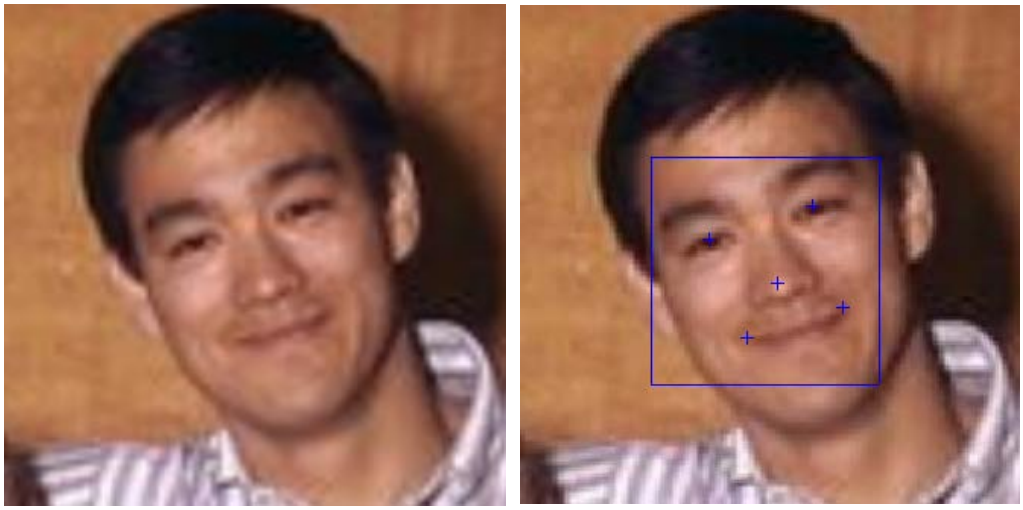


图 5-6 人脸检测和特征点检测

Fig.5-6 Face detection and face point detection

根据模板特征点和检测到的特征点，计算其仿射变换矩阵，再对齐人脸图像。图 5-6 中的图像对齐后如图 5-7 所示。因为选取的参考图像为图 5-4 中左图，因此图 5-7 中的人脸略微向左偏。

3. 图像剪切

对齐后可根据网络模型参数要求剪切成为 128×128 或者 64×64 的图像，之后统一送入模型中训练。

图 5-7 对齐后人脸图像

Fig.5-7 Face image after alignment

5.3 准确率测试方法及结果分析

5.3.1 实验方法

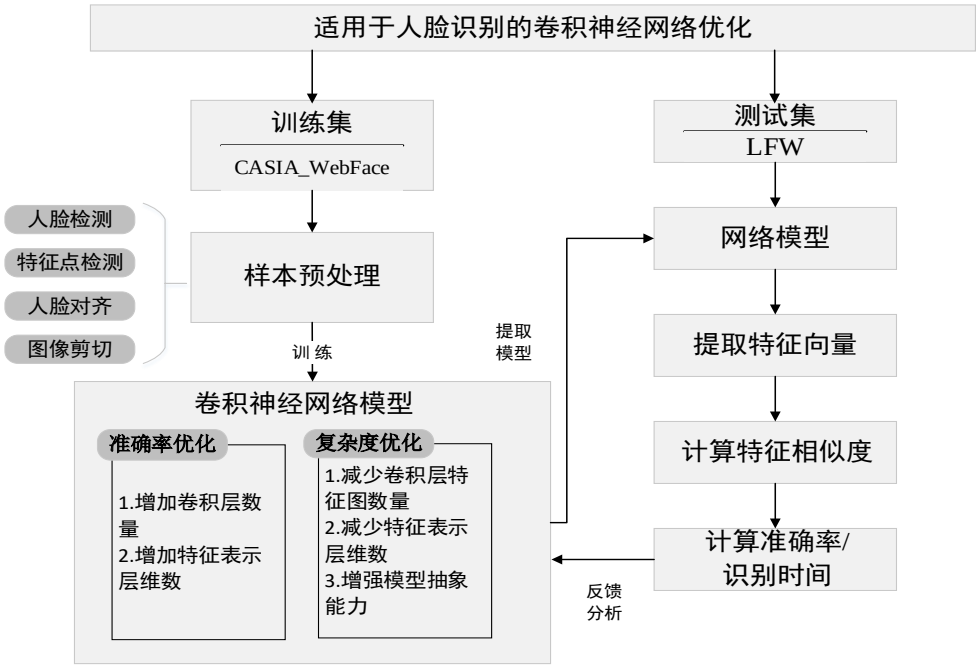


图 5-8 CNN 设计优化流程

Fig.5-8 Design and optimization flow of CNN

适用于人脸识别的卷积神经网络结构设计和优化流程如图 5-8 所示。整个流程主要分为三个部分，包括数据预处理、网络模型设计和优化以及测试流程。训练集经过预处理后输入模型进行训练，得到 CNN 模型；测试集通过训练好的模型统计准确率和复杂度（识别时间）；对准确率和复杂度进行分析，针对网络模

型再进行相应的优化。

本文采用 LFW 数据集进行测试人脸识别的准确率。从数据集中随机选择了 6000 对人脸进行测试，其中 3000 对人脸属于同一个人，另外 3000 对人脸是不同的人。测试的时候，将一对标准图像输入模型中，提取一对向量，本文中直接使用余弦相似度进行比对，系统判定这一对图片是否是同一个人。当相似度超过某个阈值时，可判定为属于同个人。本文阈值的选取采取从 0.5 增加到 0.8 的方法，每次迭代增加 0.001，计算当前阈值下的准确率。并根据阈值和准确率的对应关系画出接收者操作特征（Receiver Operating Characteristic, ROC）曲线。ROC 曲线是用于评价二元分类模型系统性能的曲线图，通过将阈值设定为连续多个不同的值，计算在此情况下的准确率，进而分析真正类率（True Positive Rate，TPR）和负正类率（False Positive Rate，FPR），拟合一条曲线，作为评价分类性能。

在二分类问题中，模型可以将输入分为正类或者负类。同时这个输入本身对应着一个正确的分类，那么这样会出现四种情况，如果一个输入是正类并且被分类为正类，即为真正类（True Positive）；如果输入是负类但被预测为正类，那么即为假正类（False Positive）；类似的还有真负类（True Negative）和假负类（False Negative），这四类如图 5-9 所示。

预测 / 实际	True	False
True	True Positive	False Negative
False	False Positive	True Negative

图 5-9 分类情况图

Fig.5-9 Classification condition

其中，真正类率 TPR 的计算公式为：

$$\mathrm{TPR} = \frac{TP}{TP+FN} \tag{5-1}$$

描述了预测到的正类占所有正样本的比例。负正类率 FPR 的计算公式为：

$$\mathrm{FPR} = \frac{FP}{TN+FP} \tag{5-2}$$

描述了预测到的负类占所有负样本的比例。

在一个二元分类模型中，假设分类的阈值为 0.7，大于这个阈值则为正类，反之为负类。如果阈值减小，那大于这个阈值的样本数会增加，因此提高了 TPR，即预测到的正类占所有正样本的比例增加。但与此同时也会增加误分类的样本数，所以也提高了 FPR。所以引入 ROC 曲线来形象地展示这一变化。

5.3.2 实验结果及分析

将相同的 LFW 测试数据集输入三种不同的模型中，利用上节提到的方法进行测试，得到的 ROC 曲线如图 5-10、图 5-11、图 5-12 所示，准确率如表 5-1 所示。

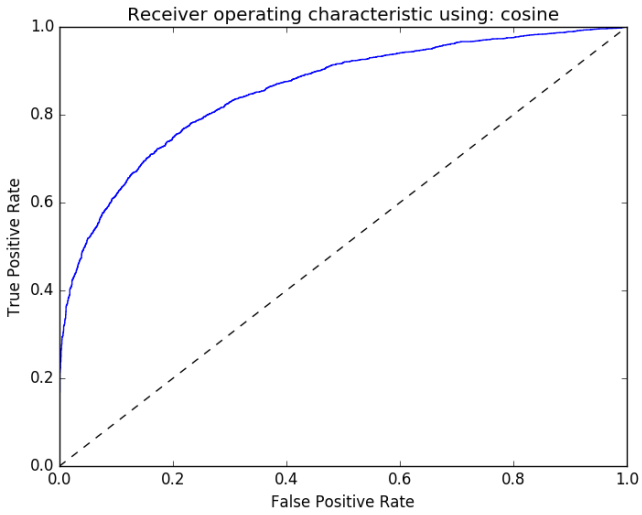


图 5-10 卷积神经网络 ROC 曲线

Fig.5-10 ROC curve of CNN model

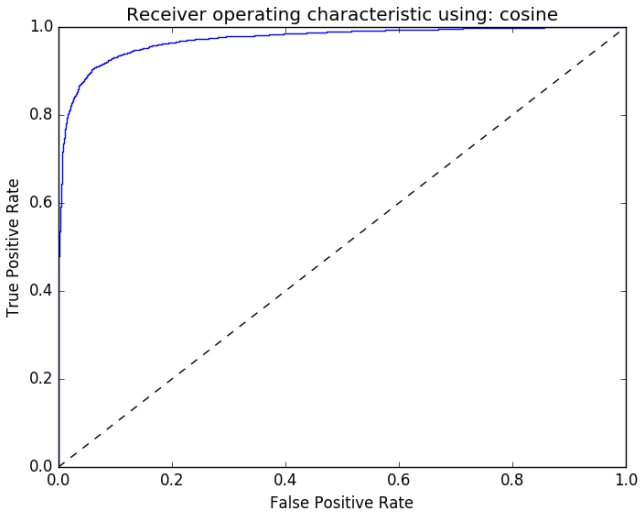


图 5-11 深度卷积神经网络 ROC 曲线

Fig.5-11 ROC curve of deep CNN model

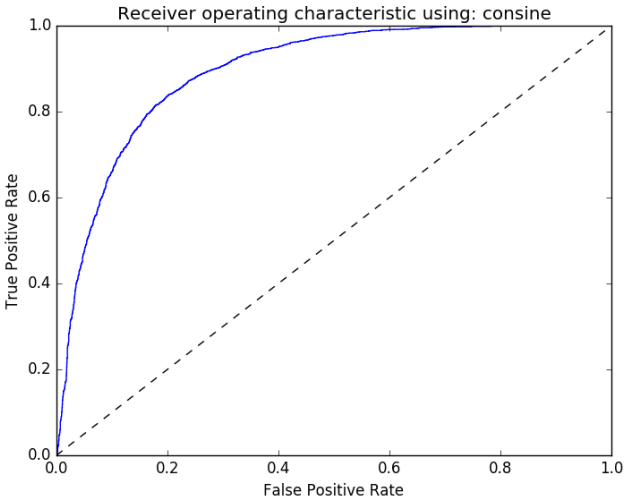


图 5-12 轻量卷积神经网络 ROC 曲线

Fig.5-12 ROC curve of lightened CNN model

表 5-1 LFW 人脸识别准确率

Table 5-1 Accuracy of face recognition

网络模型	准确率
卷积网络模型	77.5%
深度卷积模型	92.2%
轻量卷积模型	81.9%

从三个模型的 ROC 曲线和表 5-1 来看，卷积网络模型由于卷积网层数只有 5 层，层数最少，并且由于特征表示层的特征向量只有 160 维，因此对于人脸特征的刻画能力有限，因此准确率最低，ROC 曲线也最平缓。而深度网络模型有 13 层卷积和 3 层全连接层，且表示特征的向量为 4096 维，能够有效地提取到人脸的足够多的高维特征，在计算当前余弦相似度度量条件下，准确率最高，能够达到 92.2%。而轻量卷积模型通过减小网络的层数，减少特征表示层的维数，降低了模型的复杂度，但与此同时准确率也有所降低，但相比于卷积网络模型仍然有所提升。

5.4 复杂度测试方法及结果分析

5.4.1 实验方法

卷积神经网络模型的复杂度与其参数数量正相关。即卷积神经网络层数越多，每一层的特征图越多、每一层的卷积核越大，那么所需要的参数就越多。参数的数量也会直接影响到卷积神经网络模型的大小。因此从理论上来看，衡量一个模

型的复杂度可以从其参数数量来分析。从实际角度来看，模型的复杂度可以从其训练好的模型大小来判断。

卷积神经网络模型对一对图像进行识别的时间长短进行衡量，这也是比较直观衡量模型复杂度的一个标准。训练好的模型在对输入图像进行识别的时候，会先提取图像的特征。提取图像特征的过程就是神经网络前向传播的过程，直到产生一组需要的特征向量。流程图如图 5-13 所示。这个流程的时间与模型的复杂度呈正相关。

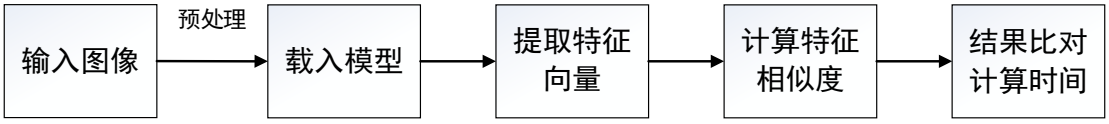


图 5-13 复杂度测试流程图

Fig.5-13 Test flow of complexity

5.4.2 实验结果及分析

根据第三章和第四章对于网络参数的分析，三种模型含有的参数以及训练后的模型大小如表 5-2 所示。

从表 5-2 中可以看出，参数数量和模型大小与模型的复杂度成正相关。卷积网络模型的参数有 550.4K 个参数，在三个模型中数量最少，因此模型大小也只有 19.7MB，也是三个模型中最小的。深度网络模型由于深度加深并且增加了维数较高的全连接层，因此参数数量有 133M 个，所对应的模型为 553MB。可以预估到，深度网络模型的识别时间会成倍于卷积网络模型的识别时间。轻量卷积模型由于使用了最大特征映射激活函数，特征图每经过一次激活层，图的数量都会减少一半，因此参数个数减少，又因为全连接层的减少和特征表示层维度的降低，因此轻量卷积模型的参数数量和模型大小都介于前两者之间。

表 5-2 模型参数及规模

Table 5-2 Model parameter and size

网络模型	参数数量	模型大小
卷积网络模型	550.4K	19.7MB
深度卷积模型	133M	553MB
轻量卷积模型	3.40M	31.2MB

表 5-3 中选取了近期比较经典的卷积神经网络模型，与本文中的轻量卷积模型进行对比。可以看出，轻量卷积模型在模型大小上有较明显优势，更适合于部署在嵌入式设备或者智能手机上。（由于准确率评价方法不一致，所以这里没有列出）

表 5-3 各模型复杂度对比
Table 5-3 Comparison of different models

网络模型	模型大小
轻量卷积模型	31.2MB
AlexNet[18]	233.0MB
GoogLeNet[29]	51.1MB
ResNet[30]	230.0MB

利用上节的实验方法，选取相同的一对人脸图输入模型中，计算识别时间，如表 5-3 所示。服务器所使用的 CPU 是 Intel(R) Xeon(R) CPU E5-2698 v3 @2.30GHz，64 核，内存 132G。

表 5-4 模型识别时间
Table 5-3 Test time of recognition

网络模型	识别时间
卷积网络模型	0.0327s
深度卷积模型	0.528s
轻量卷积模型	0.103s

从表 5-3 中可以看出，卷积网络模型的识别时间最短，而深度卷积模型的识别时间最长。这是由于三种网络模型的深度不同，并且不同层的参数数量也有所不同。深度卷积模型层数最多，且参数最多（533MB），尤其是在全连接层，因此前向传播的过程相比于其他两个模型都要慢很多。而轻量卷积模型虽然层数与深度卷积模型基本一致，然而由于其卷积层含有 1×1 的嵌套网络以及激活层采取最大特征映射函数，因此单个卷积层的参数和特征图的数量都大大减少，前向传播的速度加快，因此在深度差不多的情况下，轻量卷积模型比深度卷积模型快 5 倍以上。由以上分析可得，模型复杂度与其在识别过程中的时间成正相关。识别时间随着模型复杂度的增加而增加。

5.5 本章小结

本章首先介绍了 Caffe 深度学习框架的特性和基本数据结构，如何利用框架中的数据结构建立卷积神经网络。随后简要介绍了测试集以及测试样本的处理方法，包括人脸检测、人脸对齐和图像剪切等操作。针对人脸识别应用场景，本章给出了测试准确率和模型复杂度的方法和度量，对本文中涉及的三种模型进行测试，并分析了这三种模型各自在人脸识别准确率和模型复杂度上的表现及其原因，同时，给出了轻量卷积模型与其他模型复杂度的对比。测试结果验证了轻量 CNN

模型能够有效降低模型的复杂度，在模型大小相当的情况下提升模型的准确率。

第六章 总结与展望

6.1 本文工作与创新点

6.1.1 主要工作

随着计算机硬件的发展和数据量的爆发式增长，深度学习成为了国内外研究的热点，而卷积神经网络作为深度学习的经典算法，在计算机视觉领域取得了巨大的进展。在人脸识别应用中，基于卷积神经网络模型的准确率的提升往往伴随着复杂度的增加。基于此，本文主要研究了基于卷积神经网络的深度学习算法，并对传统的卷积神经网络做出了改进提升准确率，同时将嵌套网络和最大特征映射函数应用于网络模型中降低其复杂度，主要工作包括以下几个方面：

1. 调研深度学习的相关技术研究现状以及在各个领域内的应用，着重介绍在计算机视觉领域卷积神经网络的研究和进展，对应用于人脸识别的卷积神经网络模型的准确率和复杂度进行分析。
2. 介绍了传统的人工神经网络，包括感知器单元和多层感知器，以及用于训练神经网络的反向传播算法。讨论了卷积神经网络区别于人工神经网络的特点，主要包括局部连接、权值共享、池化操作，以及卷积神经网络的组成架构。
3. 针对人脸识别的应用场景，构建了含有 4 个卷积层和 1 个全连接层的深度网络模型，详细介绍了其架构设计、网络参数设计以及网络训练流程。在此基础上，为了提升人脸识别的准确率，给出了更深的网络结构，包含 13 个卷积层和 3 个全连接层，对比两个模型的网络结构和网络参数，分析了其复杂度和训练所需要的参数数量。
4. 人脸识别准确率的提升伴随着复杂度的成倍增长，为了降低复杂度，提出了基于轻量卷积神经网络的模型，利用嵌套网络抽象更好的特征来减少网络的层数，同时使用最大特征映射激活函数替代 ReLU 减少特征图的数量。通过对网络深度、参数和激活函数进行优化，明显降低模型的复杂度。
5. 简要介绍了 Caffe 深度学习框架特性及其数据结构，利用 Caffe 框架搭建并训练了本文提到的三种网络结构。训练集采用 CASIA-WebFace 数据集并对其进行预处理，包括人脸检测、人脸对齐和图像剪切。测试采用 LFW 数据集，详细介绍了准确率和复杂度的测试方法，对比分析了三种网络模型的准确率和复杂度，验证了之前对模型修改的假设。

6.1.2 创新点

本文的创新点主要包含以下几个方面：

1. 近来许多基于卷积神经网络的模型利用增加网络层数的方法用于提高人脸识别的准确率，然而这些方法包含模型的参数很多，复杂度也随着上升。本文利用尺寸 1×1 的卷积核部分替代原本模型中较大尺寸的卷积核，然后再把卷积层嵌套在网络结构中，从而减少了模型的参数，同时也提高了模型对特征的提取能力。
2. 为进一步降低网络模型的复杂度，本文对卷积层的特征图的数量进行优化。利用最大特征映射激活函数替代 ReLU 激活函数。特征图的数量每经过一次激活层都会减少为原来的一半，能够有效减少模型的参数。这在损失较小性能的情况下进一步降低了模型的复杂度。

6.2 下一步工作

卷积神经网络作为深度学习研究的热门算法，在计算机视觉中应用前进广泛。本文提出了适用于人脸识别应用的三种深度网络模型，分别在准确率或复杂度方面有所改进，但针对卷积神经网络的研究和优化仍有许多工作需要进一步完善。后续研究重点可以从以下两方面开展：

1. 进一步优化卷积神经网络参数和训练参数。根据前文的理论分析，可通过改变卷积神经网络的深度、激活函数和训练参数等来提升网络模型的准确率，降低模型的复杂度。而其中网络参数和训练参数的选择对于模型的训练速度和准确率也起着重要的作用，需要在实践过程中不断的调试和优化。
2. 输入图像通过网络模型之后得到关于人脸的特征向量，本文使用余弦相似度来衡量两个向量间的相似度。事实上，这种方法只能对两个向量之间的相似度做出大致的估算。如果向量维度比较低，就需要深入挖掘其中向量间的关系。后续研究中可以从这方面入手，使用一些机器学习的方法训练分类器来评估向量间的相似度。
3. 对于人脸分类的场景，单个网络模型提取图像的人脸特征泛华能力比较弱，不具有非常强的鲁棒性，这会导致一些分类错误。后续的工作会考虑研究集成技术在分类问题中的应用，通过在训练集中选取不同尺度的图像训练多个不同的模型，将这些模型通过随机森林等算法集成在模型中，用以提升人脸识别的泛化能力和鲁棒性。

参 考 文 献

[1] Sun, Yi, Xiaogang Wang, and Xiaoou Tang. "Deep learning face representation from predicting 10,000 classes." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014.

[2] Huang, Gary B., et al. Labeled faces in the wild: A database for studying face recognition in unconstrained environments. Vol. 1. No. 2. Technical Report 07-49, University of Massachusetts, Amherst, 2007.

[3] Liu J, Deng Y, Huang C. Targeting ultimate accuracy: Face recognition via deep embedding[J]. arXiv preprint arXiv:1506.07310, 2015.

[4] Parkhi O M, Vedaldi A, Zisserman A. Deep face recognition[C]//British Machine Vision Conference. 2015, 1(3): 6.

[5] Schroff F, Kalenichenko D, Philbin J. Facenet: A unified embedding for face recognition and clustering[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 815-823.

[6] Sun Y, Chen Y, Wang X, et al. Deep learning face representation by joint identification-verification[C]//Advances in Neural Information Processing Systems. 2014: 1988-1996.

[7] Deng L. A tutorial survey of architectures, algorithms, and applications for deep learning[J]. APSIPA Transactions on Signal and Information Processing, 2014, 3: e2.

[8] Deng L, Yu D. Deep Learning[J]. Signal Processing, 2014, 7: 3-4.

[9] Schmidhuber J. Deep learning in neural networks: An overview[J]. Neural Networks, 2015, 61: 85-117.

[10] Aizenberg I, Aizenberg N N, Vandewalle J P L. Multi-Valued and Universal Binary Neurons: Theory, Learning and Applications[M]. Springer Science & Business Media, 2013.

[11] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural computation, 2006, 18(7): 1527-1554.

[12] Hinton G E. Learning multiple layers of representation[J]. Trends in cognitive sciences, 2007, 11(10): 428-434.

[13] Zhu Q, Chen B, Morgan N, et al. Tandem connectionist feature extraction for conversational speech recognition[C]//International Workshop on Machine Learning for Multimodal Interaction. Springer Berlin Heidelberg, 2004: 223-231.

[14] Mohamed A, Dahl G E, Hinton G. Acoustic modeling using deep belief networks[J]. IEEE Transactions on Audio, Speech, and Language Processing, 2012, 20(1): 14-22.

[15] Robinson A J. An application of recurrent nets to phone probability estimation[J]. IEEE transactions on Neural Networks, 1994, 5(2): 298-305.

[16] Maas A, Le Q V, O'neil T M, et al. Recurrent neural networks for noise reduction in robust ASR[J]. 2012.

[17] Graves A, Mohamed A, Hinton G. Speech recognition with deep recurrent neural networks[C]//2013 IEEE international conference on acoustics, speech and signal processing. IEEE, 2013: 6645-6649.

[18] Krizhevsky A, Sutskever I, Hinton G E. Imagenet classification with deep convolutional neural networks[C]//Advances in neural information processing systems. 2012: 1097-1105.

[19] Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2014: 580-587.

[20] Lin K, Yang H F, Hsiao J H, et al. Deep learning of binary hash codes for fast image retrieval[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops. 2015: 27-35.

[21] Mikolov T, Chen K, Corrado G, et al. Efficient estimation of word representations in vector space[J]. arXiv preprint arXiv:1301.3781, 2013.

[22] Hubel D H, Wiesel T N. Receptive fields and functional architecture of monkey striate cortex[J]. The Journal of physiology, 1968, 195(1): 215-243.

[23] Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position[J]. Biological cybernetics, 1980, 36(4): 193-202.

[24] Le Cun B B, Denker J S, Henderson D, et al. Handwritten digit recognition with a back-propagation network[C]//Advances in neural information processing systems. 1990.

[25] LeCun Y, Bottou L, Bengio Y, et al. Gradient-based learning applied to document recognition[J]. Proceedings of the IEEE, 1998, 86(11): 2278-2324.

[26] Hecht-Nielsen R. Theory of the backpropagation neural network[C]//Neural Networks, 1989. IJCNN., International Joint Conference on. IEEE, 1989: 593-605.

[27] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European Conference on Computer Vision. Springer International

Publishing, 2014: 818-833.

[28] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.

[29] Szegedy C, Liu W, Jia Y, et al. Going deeper with convolutions[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1-9.

[30] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[J]. arXiv preprint arXiv:1512.03385, 2015.

[31] Taigman Y, Yang M, Ranzato M A, et al. Deepface: Closing the gap to human-level performance in face verification[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2014: 1701-1708.

[32] Taigman Y, Yang M, Ranzato M A, et al. Web-scale training for face identification[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 2746-2754.

[33] Sun Y, Wang X, Tang X. Deeply learned face representations are sparse, selective, and robust[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 2892-2900.

[34] Denton E L, Zaremba W, Bruna J, et al. Exploiting linear structure within convolutional networks for efficient evaluation[C]//Advances in Neural Information Processing Systems. 2014: 1269-1277.

[35] Fang H, Gupta S, Iandola F, et al. From captions to visual concepts and back[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015: 1473-1482.

[36] Iandola F N, Moskewicz M W, Ashraf K, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 1MB model size[J]. arXiv preprint arXiv:1602.07360, 2016.

[37] Rosenblatt F. The perceptron, a perceiving and recognizing automaton Project Para[M]. Cornell Aeronautical Laboratory, 1957.

[38] Mitra S, Pal S K. Fuzzy multi-layer perceptron, inferencing and rule generation[J]. IEEE Transactions on Neural Networks, 1995, 6(1): 51-63.

[39] Ström N. Sparse connection and pruning in large dynamic artificial neural networks[C]//EUROSPEECH. 1997.

[40] LeCun Y, Jackel L D, Bottou L, et al. Learning algorithms for classification: A comparison on handwritten digit recognition[J]. Neural networks: the statistical mechanics perspective, 1995, 261: 276.

[41] Nair V, Hinton G E. Rectified linear units improve restricted boltzmann

machines[C]//Proceedings of the 27th International Conference on Machine Learning (ICML-10). 2010: 807-814.

[42] Sermanet P, LeCun Y. Traffic sign recognition with multi-scale convolutional networks[C]//Neural Networks (IJCNN), The 2011 International Joint Conference on. IEEE, 2011: 2809-2813.

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

[44] CASIA-WebFace, http://www.cbsr.ia.ac.cn/english/CASIA-WebFace-Database.html

[45] Lin M, Chen Q, Yan S. Network in network[J]. arXiv preprint arXiv:1312.4400, 2013.

[46] Wu X, He R, Sun Z. A Lightened CNN for Deep Face Representation[J]. arXiv preprint arXiv:1511.02683, 2015.

[47] Bengio Y, Courville A, Vincent P. Representation learning: A review and new perspectives[J]. IEEE transactions on pattern analysis and machine intelligence, 2013, 35(8): 1798-1828.

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

[49] Maas A L, Hannun A Y, Ng A Y. Rectifier nonlinearities improve neural network acoustic models[C]//Proc. ICML. 2013, 30(1).

[50] He K, Zhang X, Ren S, et al. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 1026-1034.

[51] Xu B, Wang N, Chen T, et al. Empirical evaluation of rectified activations in convolutional network[J]. arXiv preprint arXiv:1505.00853, 2015.

[52] Jia Y, Shelhamer E, Donahue J, et al. Caffe: Convolutional architecture for fast feature embedding[C]//Proceedings of the 22nd ACM international conference on Multimedia. ACM, 2014: 675-678.

致　谢

攻读硕士学位期间已发表或录用的论文

[1] 第一作者, An Accurate Eye Pupil Localization Approach based on Adaptive Gradient Boosting Decision Tree[C]//Visual Communications and Image Processing Conference, 2016 IEEE