

大连理工大学本科毕业设计（论文）

基于深度学习方法的古建筑砌体结构表层损伤 识别与定位

**Superficial Damage Identification and Localization for Masonry
Historic Structures Based on Deep Learning Method**

学院（系）： 建设工程学部

专业： 土木工程

学生姓名： 赵庆安

学号： 201351043

指导教师： 赵雪峰 教授

副导师： 赵鹏 高级工程师

评阅教师： 李冬生 副教授

完成日期： 2017.06.03

大连理工大学

Dalian University of Technology

摘要

砌体结构是古建筑最重要的组成部分之一，其损伤主要体现为强度、刚度以及整体性的下降。造成结构的损伤，除了材料老化，施工质量以及本身结构体系等内部因素的影响，很大程度上在于包括物理风化、生物化学腐蚀、地震和战争破坏在内的外部因素，这些影响主要体现在结构的表层，如裂缝、酥碱、剥落、倾斜、空鼓等等。因此，对这些表层损伤实施快速和高效的识别和定位，为后续的修复、维护和管理工作提供指导具有重大的意义。

目前世界范围内对于古建筑砌体结构表层损伤的检测主要为人工方法，即通过目测和借助专业的设备完成工作。然而人工方法效率不高，且需要较强的专业性和一些经验因素。基于计算机图像处理方式的损伤检测研究也有很多，但这些研究需要大量的特征提取，识别效率和效果均具有局限性，且在噪声较多的环境条件下鲁棒性很差。

为了解决上述问题，本文提出了一种基于深度学习方法的古建筑砌体结构损伤检测技术。利用深度学习的好处在于，只要提供足够的样本，计算机可以在大量的数据训练中自动学习到结构损伤的识别方法，无需人工的特征提取技术。一旦模型训练完成，可以利用其直接检测损伤。检测工作的进行几乎无需任何专业性和经验性，且不受环境因素的影响，从而达到高效和准确的目的。本文以北京故宫城墙结构作为研究和实验对象，所做的创新性工作如下：

1. 基于深度学习方法，利用卷积神经网络，训练出一个能够自动识别砖块单位损伤的分类器。对于输入的单张砖块单位图像，分类器能够识别该砖块单位是否受到表层损伤，若有损伤，能够识别该损伤为裂缝、酥碱或是剥落的哪一种。
2. 基于一种滑动窗口的算法，结合训练好的分类器，提出了一种砌体结构表层损伤定位技术。对于输入的结构表层图像，能够定位到各个损伤的砖块单位，并能够对损伤进行分类。
3. 为了进一步提高检测效率，提出了一种基于候选区域目标的深度学习损伤检测技术，对于输入的结构表层图像，能够准实时地识别定位损伤砖块单位。
4. 初步提出了一种众包式损伤检测模式，通过调动公众实现样本数据的高效大量收集，为后续工作提供极大的便利，使得基于深度学习的损伤检测技术在实际工程中的广泛应用成为可能。

关键词：古建筑；砌体结构；深度学习；表层损伤；检测

Superficial Damage Identification and Localization for Masonry Historic Structures Based on Deep Learning Method

Abstract

Masonry structures, which subjected to various forms of damage, mainly resulting in the reduction of strength, stiffness and integrity, are one of the core components of historic architecture. The factors causing the structural damage, in addition to internal ones such as materials aging, construction quality and structural systems, are most external ones including physical, chemical and biological weathering, earthquakes and wars. Such damage always reflects on the surface of structures, such as crack, efflorescence, spall, incline, hollowness and so forth. Hence, it is of great significance to identify and localize the superficial damage quickly and efficiently. The work can be regarded as guidance for structural repair, maintenance and management afterwards.

Artificial inspection (i.e., visual inspection and/or with professional equipment) is the most common approach in the world to identifying and assessing superficial damage at present. However, such method is inefficient, heavily relying on professionals and their experience. Plenty of research on image processing techniques-based damage detection has been implemented. However, such method requires extensive feature extractions, which limits the efficiency and performance. The robustness is insufficient under the high levels of ambient noise conditions as well.

To overcome those limits, in this thesis, a deep learning-based damage detection technique for masonry historic structures is proposed. The advantage of deep learning method is that with sufficient training samples, machines are able to learn how to detect the damage automatically via training without using artificial feature extraction techniques. Once the model training is complete, it can be used to detect damage directly. The detection requires hardly any professional abilities or experience, and is hardly affected by the environment. Hence, high efficiency and accuracy can be realized. The wall structures of Beijing Imperial Palace are regarded as research and experiment objects in this thesis. The innovative works are presented as follows:

1. A deep learning-based damage classifier is trained using convolutional neural networks. For the input images of single brick unit, the classifier is able to determine whether the brick is damaged. If the answer is true, the classifier can identify 3 damage categories (crack, efflorescence and spall).

2. A sliding window algorithm-based superficial damage localization technique using the trained classifier is proposed. For the input images of the structures, damaged brick units can be localized and classified.
3. To further improve the detection efficiency, a region proposal-based deep learning damage detection technique is proposed. For the input images of the structures, damaged brick units can be detected directly in quasi real-time.
4. A crowdsourcing damage detection mode is preliminarily proposed. Sample data can be efficiently collected in large numbers by giving rise to public initiatives. Such mode can provide great convenience for follow-up work, which makes the deep learning-based damage detection technique widely used in practical engineering possible.

Key Words: Historic Architecture; Masonry Structures; Deep Learning; Superficial Damage; Detection

目 录

摘要	I
Abstract	II
1 文献综述	1
1.1 研究背景意义	1
1.2 相关研究工作概述	3
1.2.1 当前工程中砌体结构检测概述	3
1.2.2 基于图像处理方法的结构表层损伤检测研究现状	4
1.2.3 基于机器学习方法的结构表层损伤检测研究现状	5
1.2.4 深度学习与土木工程结构检测	6
1.3 当前存在的问题	8
1.4 本文研究内容	9
2 基于卷积神经网络的砖块表层损伤分类技术	11
2.1 方法概述	11
2.2 基本原理	11
2.2.1 图像分类	11
2.2.2 神经网络	16
2.2.3 卷积神经网络	20
2.3 网络架构及超参数配置	26
2.3.1 AlexNet for HMDI 架构及超参数配置	26
2.3.2 GoogLeNet for HMDI 架构及超参数配置	27
2.4 本章小结	31
3 建立数据集及准备训练环境	32
3.1 数据集的建立	32
3.2 软硬件环境配置	34
3.3 Caffe 深度学习框架的搭建	35
3.3.1 Caffe 简介	35
3.3.2 Caffe 的编译	36
3.3.3 卷积神经网络在 Caffe 中的实现	37
3.4 本章小结	39
4 基于滑动窗口算法的损伤识别与定位技术	40
4.1 损伤识别技术	40

4.1.1 二分类实验.....	41
4.1.2 AlexNet for HMDI 四分类实验	44
4.1.3 GoogLeNet for HMDI 四分类实验.....	47
4.2 损伤定位技术.....	50
4.2.1 滑动窗口算法.....	50
4.2.2 实验验证.....	52
4.3 同数字图像处理方法的对比研究.....	55
4.4 本章小结	57
5 基于候选区域目标的深度学习损伤检测技术.....	59
5.1 方法原理	59
5.1.1 RCNN 和 Fast-RCNN.....	59
5.1.2 Faster-RCNN.....	62
5.2 基于 Faster-RCNN 的损伤检测实验.....	65
5.3 本章小结	69
6 众包式古建筑损伤检测模式初探.....	70
6.1 众包式古建筑损伤检测模式	70
6.2 本章小结	71
结 论	72
参 考 文 献	73
附录 攻读本科学位期间发表学术论文情况	80
致 谢	81

1 文献综述

1.1 研究背景意义

古建筑作为一种重要的文明载体，是世界历史文化遗产不可或缺的一部分。自 21 世纪以来，各个国家对于古建筑的保护越发重视，维护和修复工作大量展开，并且得到了长足的发展^[1-4]。以我国为例，近 60 年来，随着国民经济实力和文化素质的不断发展，国家文物保护单位近乎以指数的形式增长。由图 1.1 可知，从 1960 年 3 月第一批公布的 180 处全国重点文物保护单位，到 2013 年 5 月公布的第七批 1943 处，目前全国的重点文物保护单位共计 4000 余处^[5]。国家文物局于 2017 年 2 月正式发布的《国家文物事业发展“十三五”规划》^[6]指出，国家将制定并完善古建筑维护规程，积极开展综合保护和修缮项目。

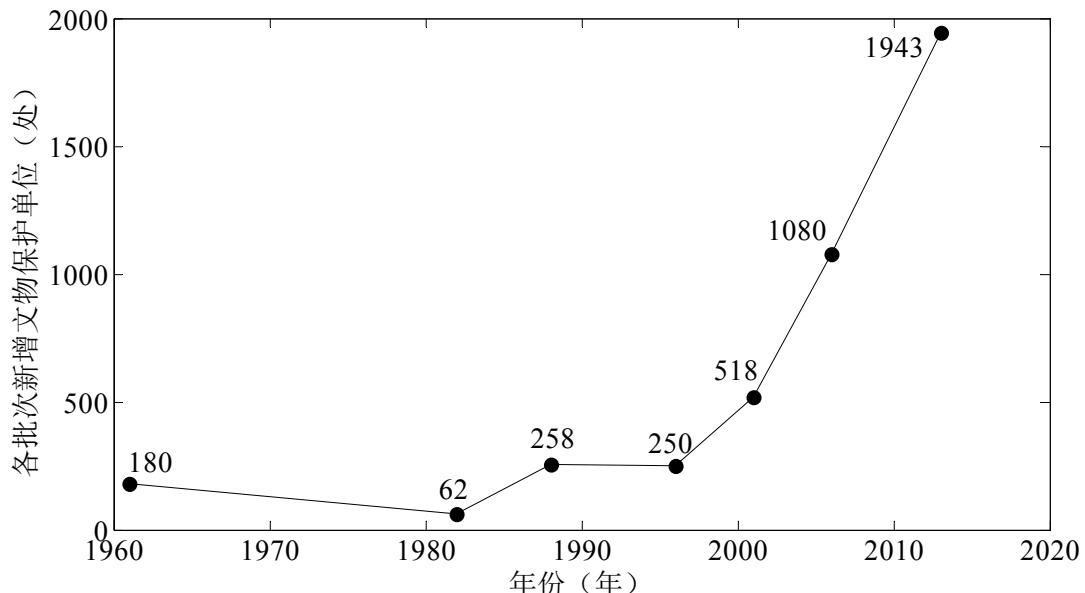


图 1.1 全国重点文物保护单位增长数量

然而，古建筑的破坏现状决定了损伤检测和维护修复工作的高难度和高成本。近些年，国内各地区大量资金投入于此方面的工作。2014 年，安徽省黄山市徽州区投入 8000 万元，对该区 147 处古建筑实施保护利用工程^[7]；同年，贵州省铜仁市投入 6000 万元，启动了 40 个古建筑维护项目^[8]；2015 年，广东省珠海市高新区投入近 3000 万元，完成了对该区 18 处重要古建筑的修缮和活化工作^[9]；2016 年，国家文物局投入 2610 万元开

始对永乐宫实施整体维修^[10]。大量高投入的维护工作对损伤检测与评定环节提出了更高的要求。

砌体结构是世界范围内现存古建筑最重要的组成部分之一，主要以具有承重和维护功能的砖石墙体形式存在，砖墙的安全性对古建筑的维护工作具有极其关键的作用^[11,12]。在长期外部因素和内部因素共同作用下，结构产生不同程度的损伤，亦称病害，如图 1.2 所示。这些病害大致分为三类^[13]，第一类为环境病害，主要由包括地震、滑坡和不均匀沉降在内的地质灾害以及包括洪水、飓风、暴雨在内的气象灾害造成；第二类为结构病害，主要由包括受压、受剪和受拉在内的力学破坏以及失稳破坏造成；第三类为材料病害，主要由材料的自然劣化、冻融、矿物溶解和生物化学腐蚀造成。这些病害很大程度上通过结构的表层损伤体现出来，如裂缝、剥落、酥碱、变形等等。尽管表层状态可能无法完全体现出结构的损伤情况，但可以通过表层损伤检测得出初步的结论，之后配合无损检测技术，最终深入分析得出结论^[14]。然而，由于人员和仪器的限制，深入内部的检测是不可能在结构的全部区域大范围进行的，只能采取关键点抽样检测的方式^[15]。同时，由于砌体结构的特殊性，大范围表层检测的优势在于可以直接为砖块的替换和修复工作提供指导。因此，作为结构检测的重要组成部分，表层检测新技术的实现，是古建筑维护工作，甚至整个土木工程结构健康监测领域的一个强有力的补充^[16]。定期进行高效的表层检测，诊断并评估损伤情况，为古建筑的修复和管理提供必要的依据和指导具有重大的意义。



图 1.2 北京故宫养心殿砖墙病害示例

1.2 相关研究工作概述

1.2.1 当前工程中砌体结构检测概述

目前在工程领域，砌体结构的检测主要包括砖强度检测、砌体砂浆强度检测、结构强度检测、构件变形的检测、结构沉降及倾斜检测、缺陷检测^[15,17]。

强度的检测方法主要包括有损检测和无损检测，有损检测主要包括原位轴压法、扁顶法、原位单剪法、筒压法等等；无损检测主要包括回弹法和射钉法。结构强度目前还无法通过对结构的表面的视觉观测直接得出结论，即便是无损检测技术，也需要得到结构的一些物理性质。例如回弹法是通过锤击结构表面后能量的变化来得出结构的强度，是基于结构表面硬度和其强度之间的联系的一种无损检测技术^[18]。因此，强度检测不在本文的讨论范围之内。

砌体结构另一类检测项目是变形及位移检测，包括构件的变形以及结构整体的沉降和倾斜，这类项目最常见的检测方式是使用水准仪和经纬仪，通过几何关系直接得到检测结果，原理简单，然而效率较低^[15,17]。其它的主要方式包括位移传感器及其分布式网络检测^[19]，全球定位系统（GPS）检测^[20]，应变传感技术检测^[21]，以及近几年迅速发展的基于数字图像处理的变形及位移检测技术^[22-25]。目前有少量的基于机器学习领域的支持向量机（SVM）算法的结构变形研究^[26,27]，而基于深度学习算法的相关研究几乎没有。本文的检测技术暂时不涉及到这一类项目。

最后一类检测项目即为缺陷检测，亦为本文所研究的重点。此类损伤大多在结构表层体现出来，检测工作也主要在结构表面进行。对于古建筑砌体结构，此类项目主要包括砌体外观质量检测、砌筑质量检测、砌筑损伤检测、腐蚀检测、构造和连接检测，在实际工程检测中除了最后一项一般均采用目测法^[28]。若需检测裂缝宽度，一般使用裂缝测宽仪；若需检测腐蚀深度，一般使用锤或铲这类简单的工具去除腐蚀层，之后使用直尺测量深度^[17]。砌体外观质量检测项目主要包括砌块的尺寸、缺棱掉角、弯曲、平整度和外观色泽等等，按照《建筑结构检测技术标准》中的抽样检测时检测批的合格判定规定执行，首次抽样数目可以为 50 块砌块或砖^[15]。砌筑质量检测项目主要包括灰缝砂浆饱满度和均匀性以及组砌方式，按照《砌体工程质量验收规范》执行^[29]。砌筑损伤检测项目主要包括裂缝检测、灾害损伤检测和人为损伤检测，同时应测绘损伤的面积及分布情况^[28]。腐蚀检测项目主要包括剥落、风化、疏松等等^[17]。对于古建筑来说，由于受长期暴露在环境中，腐蚀情况往往较其他砌体结构严重很多，比如很常见的酥碱现象。“酥碱”现象指建筑材料中的可溶盐和碱随着湿度的变化溶出并结晶，聚集于结构的表面^[30]，

若不及时处理，结构的酥碱部位将逐渐变软并脱落，严重影响外观甚至安全性。以北京故宫为例，城墙青砖表面该现象尤为严重，如图 1.3 所示。



图 1.3 北京故宫城墙表面酥碱现象

1.2.2 基于图像处理方法的结构表层损伤检测研究现状

随着近年来数字信号处理技术的飞速发展，图像处理技术在无损检测领域中已占有较大的比重。通过对图像分析技术模型的深入研究和运用，该方法在无损检测中已经可以提供十分精确的数据^[31]。对于结构的表层损伤检测来说，图像处理技术的应用尤为广泛，因为几乎所有此类的损伤，例如裂缝和腐蚀，都是可以在像素中直接辨识出来的。通过对结构表面进行拍摄，将获得的数字图像进行分析处理，从而得出检测结果。

图像处理技术在结构表层损伤检测中的应用主要包括边缘检测技术^[32]、变换技术^[33]以及特征统计技术^[34]。边缘检测技术是图像处理应用于结构表层损伤检测最为广泛的技术，主要分为图像滤波、图形增强、图像检测和图像定位四个步骤^[35]。滤波的作用是提高边缘检测中对于噪声相对敏感部分的性能的提升，增强算法是用于获得相邻像素点的数值变化，此算法可用来检测像素值变化明显的区域。在实际检测的过程中，由于物体投影到二维平面的过程中信息的丢失，边缘点可能无法直接精确地检测出来。然而图像中的边缘具有方向性，即像素沿着边缘方向变化较为缓慢，垂直边缘方向变化相对显著^[36]。利用边缘的这种方向性，通过微分算子便可以将其检测出来。常见的算子有 Sobel 算子^[37]、Robert 算子^[38]、Prewitt 算子^[39]和 Canny 算子^[32]。其中 Sobel、Robert、Prewitt 均属于一阶微分的边缘检测算子，在检测的过程中使用不同大小的卷积模板与图像的像素做卷积运算，根据特定的阈值将边缘点提取出来。Canny 算子的处理方式与前几种算子有些不同，首先对图像应用高斯滤波器进行去噪和平滑处理，之后使用有限差分法(一

阶偏导) 得到梯度的方向和幅值, 接着利用梯度的方向抑制幅值的非极大值, 最后采用双阈值的方式确定边缘^[40]。

变换技术主要包括 Haar 小波变换和傅立叶变换。美国学者 Abdel-Qader 等学者曾使用这两种变换方式以及前面提到的 Canny 算子和 Sobel 算子来检测混凝土裂缝, 他通过实验验证, 认为 Haar 变换是最佳的方式^[41]。特征统计技术是通过计算图像每一个小区域的局部特征来分析图像灰度值的空间分布, 从而得到图像局部特征的统计分布^[34]。

近年来, 将图像处理技术应用于结构表层损伤检测的研究不断发展。2006 年, Corr 等学者利用数字图像技术分析了钢筋混凝土界面脱粘和碎裂情况^[42]。2007 年, Falsone 和 Lobardo 开发了一项基于图像处理的砌体结构表层随机特征分析技术^[43]。2009 年, Kabir 等学者利用大量的图像处理技术对混凝土大坝的表层损伤进行检测和研究^[31]。2013 年, Ghiassi 等学者应用数字图像相关法, 成功分析了砌块和纤维增强复合材料的粘接性能^[44]; Genceturk 等学者应用类似的方法对预应力混凝土结构全尺寸的检测进行了一系列的研究^[45]。同年, Cavalagli 等学者通过图像处理技术对砌体结构表面的砌块和砂浆的几何参数进行统计分析, 获得其概率分布并识别纹理类型^[46]。2014 年, Nejad 等学者利用图像处理方法提取了沥青混凝土的表面特征, 并开发了一个应用于特征识别和检测的算法^[47]。2015 年, Mahal 等学者应用数字图像相关法对钢筋混凝土梁的疲劳特性进行了研究^[48]。2016 年, Hamrat 等学者利用数字图像技术研究了纤维混凝土的弯曲开裂情况^[49]。

1.2.3 基于机器学习方法的结构表层损伤检测研究现状

自进入 21 世纪以来, 人工智能领域的持续升温。作为人工智能的核心, 机器学习技术及其下属的各个方向均在不断地发展。简单的说, 机器学习就是研究如何使计算机能够像人类一样从数据中进行学习, 从而获得分析和预测数据的能力。按学习的形式主要分为监督学习、无监督学习和半监督学习。目前的研究与应用尚以监督学习为主, 监督学习的主要任务是分类和回归, 即分析和预测数据^[50]。本文所研究的所有内容均属于监督学习。

机器学习应用到结构表层损伤检测的研究有很多, 主要应用的相关技术包括人工神经网络、SVM 和 k 邻近算法等等。但机器学习的应用主要是现有无损检测技术的一个附属工具, 即先收集各个传感器的信号, 之后利用机器学习技术判定检测信号是否能够表明结构的损伤以及损伤情况。例如 2002 年, Liu 等学者利用神经网络技术对混凝土无损检测结果进行分析, 实现了裂缝类型、位置和长度的检测^[51]; 2007 年, Jiang 和 Adeli 基于小波神经网络提出了一种叫做“伪谱”的损伤检测技术, 他们使用一个 38 层混凝土结构模型的数据进行分析和验证, 实现了利用少量的传感数据就可以检测出损伤的目

的^[52]；2014年，Sipos 和 Sigmund 等学者应用神经网络和主成分分析技术提出了一种可以预测框架砌体结构单元抗震性能的方法^[53]；同年，Butcher 等学者基于随机神经网络对钢筋混凝土表层损伤检测进行了深入研究，应用了一种叫做“极限学习机”的算法，检测效果较传统方式提高很多^[54]。

与此同时，许多对于结构表层损伤的研究将机器学习技术与图像处理技术结合在一起，即先由图像处理技术提取特征，再由机器学习技术进行分类。例如 2010 年，Kabir 应用灰度共生矩阵特征分析和人工神经网络分类器对混凝土碱骨料反应诱发的裂缝进行了检测^[55]；2011 年，Moon 和 Kim 应用滤波技术和反向传播（BP）神经网络对混凝土裂缝进行检测，达到了很好的精度^[56]；同年，Jahanshahi 等学者基于三维场景重建、图像处理、神经网络和 SVM 技术实现了混凝土裂缝的检测和量化^[57]；2013 年，Byrne 等学者基于灰度共生矩阵和 SVM 提出了一种半自动的增强纹理分割方法，用来检测和分类基础设施结构的表层损伤^[58]；2014 年，Plevris 和 Asteris 应用神经网络对砌体结构在双轴压应力作用下产生的表面损伤进行了建模分析^[59]；Byrne 等学者基于 Sobel 边缘检测和 SVM 提出了一种区域增强多相分割技术，用来检测基础设施的多种表层损伤^[60]；Wu 等学者结合图像处理技术和人工神经网络分类器，并提出了一个叫做“MorphLink-C”的技术，用于裂缝特征片段的聚类和连接，大大提高了检测精度^[61]。

1.2.4 深度学习与土木工程结构检测

深度学习由机器学习领域的领军人物 Geoffrey Hinton 于 2006 年正式提出^[62]，作为机器学习领域的一个重要分支，深度学习是当前人工智能领域研究和开发的热点。相对于 SVM 算法等浅层学习，深度学习中的数据特征无需通过人工设计的方式提取，而是通过计算机的自动学习来构建最佳的分类模型^[63]。其中应用最广泛，研究最热点的学习模型即为卷积神经网络。可以说，深度学习之所以可以做得很“深”，正是通过卷积技术来实现的，卷积神经网络使得深度学习的性能与效果几乎优越于机器学习所有其他的方向。本文研究基于的深度学习方法，即为卷积神经网络。

卷积神经网络由最初的浅层人工神经网络发展而来。1962 年 Hubel 和 Wiesel 受猫的视觉神经启发，首次提出了“感受野”的概念^[64]，这可以看作是卷积神经网络的雏形。1986 年，Rumelhart 等学者首次提出了误差反向传播算法，大大降低了神经网络的计算难度，使其效果完全超越人工神经网络最初提出的“感知器”算法^[65]，重新使神经网络的研究成为热点。1989 年，“卷积神经网络之父” Yann LeCun 将神经网络反向传播算法应用于手写邮政编码的识别^[66]。1998 年，LeCun 首次将卷积神经网络应用于手写数字识别，设计出了第一个卷积神经网络模型，这就是著名的 LeNet^[67]。直到今天，许多图像的分类任务仍然在使用这个模型。一般来说，尤其对于图像识别任务，网络层数越

多越好，但是当时由于训练数据的缺乏以及硬件计算速度的局限使得无法发展更深的网络，而浅层的卷积神经网络相比于 SVM 没有太大的优势。深度学习真正成为当前的研究热点，需要归功于 ImageNet 图像识别挑战赛（ILSVRC）^[68]。ImageNet 项目^[69]由斯坦福大学计算机系建立，是当前最大的图像识别数据库，ILSVRC 的主要任务是利用自己设计的模型对 1000 类物体进行图像识别，准确率高者获胜。随着大数据和使用算机图像处理单元（GPU）高速训练的实现，卷积神经网络再次掀起高潮。2012 年，前文提到的 Geoffrey Hinton 和他的研究生 Alex Krizhevsky 首次使用卷积神经网络参加 ILSVRC，将前五类识别错误率由 2011 年冠军模型的 25% 降到 16%，这就是著名的 AlexNet^[70]，从此深度学习从机器学习中脱颖而出，成为研究和应用的热点。2012 年之后的模型，如 ZFNet^[71]、GoogleNet^[72]、VGG-Net^[73]都是在 AlexNet 的基础上进行改进和加深。直到 2015 年，微软亚洲研究院重新设计了网络中的信息流动，提出“残差学习”的概念，并设计出 ResNet 模型，将前五类错误率降到 3.57%，而人眼识别的错误率是在 5.1% 左右^[74]。2016 年，谷歌公司将 GoogLeNet 中的 Inception 技术与“残差学习”结合，将前五类错误率降到 3.08%^[75]。2010 年-2016 年 ILSVRC 冠军模型前五类分类错误率如图 1.4 所示。

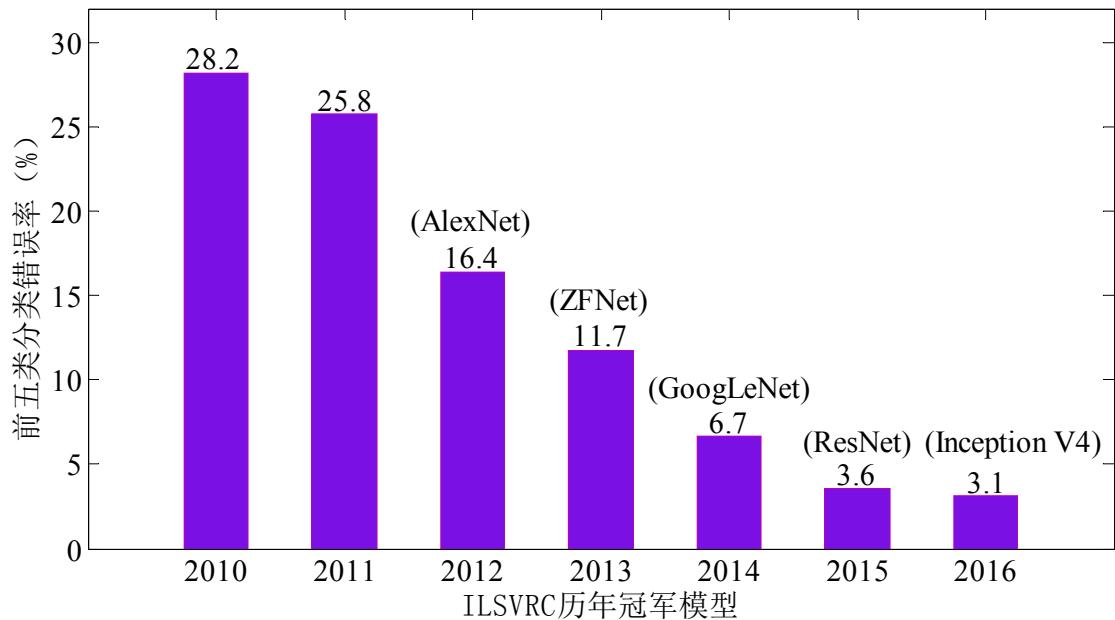


图 1.4 2010-2016 ILSVRC 历年冠军模型前五类分类错误率

然而，深度学习应用于土木工程领域的研究很少，原因之一就是需要海量的数据支持，然而在分类数目较少的情况下还是较为容易实现的。目前所有的相关研究几乎均为

结构损伤研究的二分类问题，使用的卷积神经网络模型大部分都是 LeNet。2004 年，Ouellette 等学者使用卷积神经网络结合遗传算法进行裂缝识别，但精度不是很高^[76]；2015 年，Makantasis 等学者在隧道混凝土表面的检测中应用了一个只有两个卷积层和一个全连接层的卷积神经网络，但在第一步还是利用了图像处理技术进行特征提取^[77]；2016 年，Protopapadakis 等学者将卷积神经网络与 3D 建模技术结合进行隧道表面裂缝的监测^[78]；同年，Abdeljaber 等学者将传感器信号喂入一维的卷积神经网络，实现基于振动法的结构实时监测，并降低了大量的计算复杂度^[79]。2017 年，Cha 等学者提出了基于卷积神经网络的混凝土裂缝完整检测方法，且达到了很高的精度^[80]。

1.3 当前存在的问题

由本文 1.2.1 部分可知，当前古建筑砌体结构的表层损伤检测一般采用以目测为主，检测设备为辅的人工方法。这种方式虽然看起来简便，但是对检测人员的专业性要求十分高，且需具备一定的损伤判定经验。例如对墙体裂缝的检测，检测人员必须具备能够熟知该裂缝的性质以及判定是否需要对其修复处理的能力。并且大范围的检测消耗人力成本很高，只能采取定期检测和抽样检测的方式。虽然就目前来说，这种检测方式可以达到合理的效果，但是有时仍然不能及时的发现问题，从而无法对维护工作提供实时和必要的指导，造成维修成本的大量提高。

若建立实时结构健康监测系统，通过各种传感器收集数据并进行实时监控，又会增加大量的成本，且仍然无法解决根本的大范围适用性问题，因为结构健康监测系统是不可能安装在每一个结构上的。即使某古建筑安装了大量的传感设备，也不可能监测每一个构件，每一处结构表面。同时，使用大量的传感设备同样需要大量的专业人员进行操作，且工作较为复杂。因此，这种方式只可能应用于特定的重要程度很高的标志性古建筑上面，而一般的古建筑只能主要借助于人工检测。

另外一个问题就是损伤的统计问题，人工检测时检测到损伤后只能以记录的方式收集数据，然后通过输入计算机进行分析和处理。这种方式有可能会造成实际与记录的偏差，从而无法得出准确的结论。即使检测人员足够仔细，且不考虑工作疲劳导致准确率下降的因素，统计工作效率仍然是不高的。例如统计一面墙有多少块砖损伤严重需要替换，以及有多少块砖只需要表面做特定的处理，这种方式只能通过人工计数的方式来进行，其效率可想而知。

一种很好的解决问题的方式就是利用机器视觉的方式完成这项工作，因为既然人可以直接通过结构表面得出结论，那么通过拍摄或扫描的方式直接将其交给计算机是一种较为可行的方案。同时，拍摄的方式效率很高，通过缩短检测周期基本可以达到“实时”

的效果，且基本能够全范围扫描结构表面，相对于结构健康监测系统的布设成本较低。因此，如本文的 1.2.2 部分所述，近些年出现了大量基于数字图像处理的结构表层损伤研究。然而这些以边缘检测为主的图像处理技术，需要对结构的损伤进行特征提取，其最大的缺陷就是对包括光照和阴影在内的噪声敏感性较大，实际情况下误判的可能性很大。常见的解决方式是应用去噪技术，然而去噪技术是在已知噪声信息很多的情况下才能够达到效果，并不具备广泛的适应性。因此，由于这种在复杂环境下鲁棒性较差的因素，图像处理方法检测表层损伤并没有在实际工程中特别广泛地应用。

为了能够提高图像检测的准确性，如本文 1.2.3 部分所述，很多的研究中引入了机器学习方法，然而基于浅层神经网络和 SVM 技术的检测仍然需要特征的提取，因此检测的鲁棒性和适应性并没有很太大的提高，并且若不借助图像处理技术，仍然需要复杂的传感设备输入信号。

深度学习的一个优势就是无需特征的提取，只要样本足够，经过训练的分类器可以准确地识别损伤。随着该研究领域的不断升温，每年都会出现许多训练更高效，分类更准确的卷积神经网络模型，将深度学习应用于结构损伤检测领域是一个必然的趋势。

目前将深度学习引入结构表层损伤检测的研究很少，相关研究的网络深度相对较浅，基本上均为二分类问题，且在识别的算法上效率不高，没有任何研究应用基于候选区域的深度学习目标检测法。同时，也尚未见到任何应用于古建筑或砌体结构的相关研究。本文的研究将对这些内容进行补充，这些补充即为本文的创新点，内容概述见 1.4 部分。

1.4 本文研究内容

第一章：首先通过大量的数据和实例分析了古建筑砌体结构表层损伤检测的必要性，讨论了本文研究的背景和意义，接着简要总结了当前工程中砌体结构的常用检测方法，之后从图像处理和机器学习应用于结构表层检测、深度学习及其在土木工程领域的应用这几方面做了简要介绍，然后提出了当前存在的问题，最后概述全文的研究内容。

第二章：概述了古建筑砌体结构表层损伤检测的基本原理。从单层线形分类器到一般的 BP 神经网络，再到卷积神经网络，全面搭建了本文的理论框架。同时，详尽说明了在网络训练过程中的计算和优化方法，并对一些可能出现的问题及其解决方式进行了分析与讨论。最后对应用于本文研究的两个卷积神经网络模型进行了架构分析和超参数配置。

第三章：详细说明了用于训练、验证和测试的样本获取，以及数据集的建立方式。同时，介绍了网络训练所用到的工作站的软硬件配置，以及 Caffe 深度学习框架的搭建。最后详解了卷积神经网络在 Caffe 中的实现。

第四章：提出了基于滑动窗口算法的损伤识别与定位技术，以北京故宫城墙为例，利用第三章建立的数据集进行网络训练，研究分类数目、数据量和网络深度对于识别效果的影响，并成功训练了一个验证准确率较高的分类器。之后使用这个分类器并基于滑动窗口算法实现了损伤的定位，并对实验结果进行分析讨论。最后将此方法同图像处理边缘检测方法对比，说明本文方法的准确性和鲁棒性。

第五章：提出了基于候选区域目标的深度学习损伤检测法，该方法很大程度上消除了滑动窗口算法的效率和适应性等方面的问题。基于 Faster-RCNN 算法，利用第三章建立的数据集进行训练和测试，并对实验结果进行分析和讨论。最后提取训练好的模型对若干另外的样本进行了损伤检测并分析结果，并对其中的问题提出了可能的解决方案。

第六章：初步提出了一个众包式损伤检测模式，通过调动公众来提高样本数据量，有望建立“结构损伤大数据”，更大程度上实现提高效率，节约成本的目的。

2 基于卷积神经网络的砖块表层损伤分类技术

2.1 方法概述

针对第一章提出的人工检测以及大型传感器系统的局限性，本文使用计算机图像视觉的方式来解决这些问题。不同于数字图像处理技术，本文基于深度学习的方法，无需对样本特征进行提取，可以解决图像处理法存在的鲁棒性和适应性缺陷，达到更高的精度和识别效率。

本文所研究的重点在于两部分：识别和定位。“识别”从根本上来说是一个“分类”问题，即输入一个砖块图像样本，计算机要对其进行分类：“损伤”还是“未损伤”，如果“损伤”，计算机还要给出“何种损伤”的输出。对于定位，本文提出了两种解决方式：第一是基于滑动窗口的方式，使用具有单个砖块图像的尺寸的滑窗在测试样本上以一定的路线和步长进行“滑动”，最终滑窗必须扫描过测试样本的全部区域。滑窗每停在一个位置，便将该区域砖块的图像输入分类器，若判定为损伤，则返回至测试样本并标记，之后继续至下一个区域扫描。如此循环，最终测试样本上将会出现许多被标记出来的砖块，这些砖块即为分类器所判定的“损伤砖块”，从而达到了定位的目的。第二种方式不同于第一种将“识别”和“定位”分开来做的思想，而是采用一种称为“区域建议网络”的算法，并将分类、定位任务置于同一个网络中，即输入一张任意尺寸的包含多个砖块的图像，可以直接自动标注损伤砖块的位置，识别和定位任务同时完成。

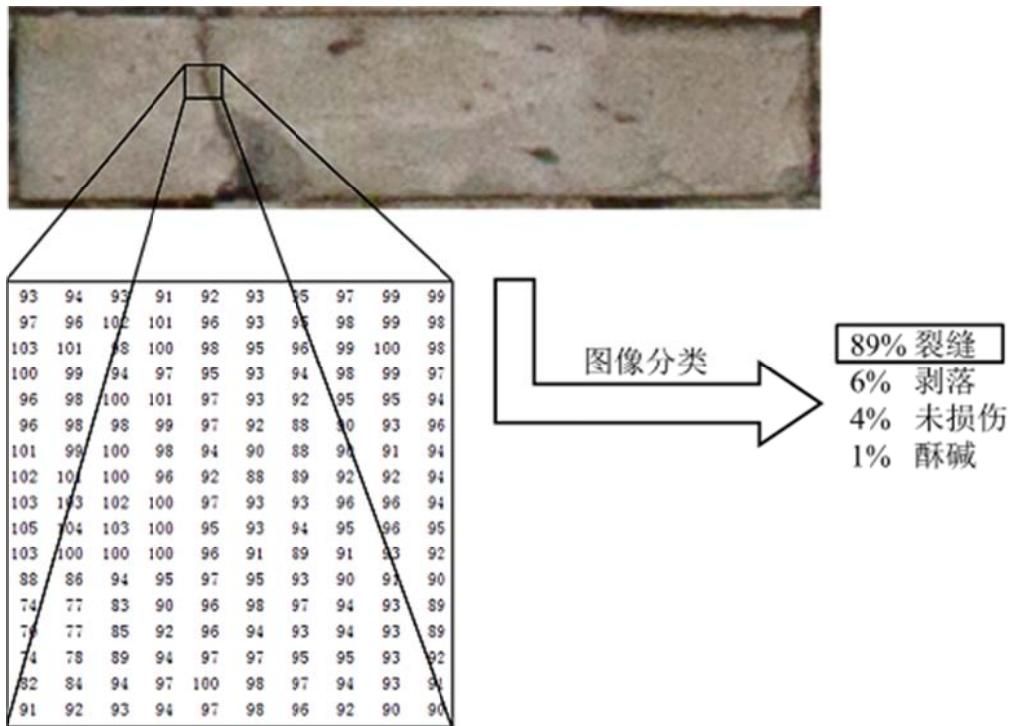
由此可知，本文的核心即为分类技术和两种定位技术。分类以及第二种定位均应用深度学习的方法。本章将结合砖块损伤的识别任务，详述分类技术的原理及实现算法。第二种定位技术原理及算法将在本文第五章详述。

2.2 基本原理

2.2.1 图像分类

以本文研究内容为例，假设输入的一张像素分辨率为 480×105 的真彩色砖块图像，如图 2.1 所示。图像以像素为单位，因此该砖块图像宽度为 480 个像素，高度为 105 个像素。由于是真彩色图像，每个像素具有红、绿、蓝三原色通道，即三个数字分量，每个分量的范围为 0-255，通过这三个分量的组合，形成了每个像素的颜色效果（例如三个分量均为 0 则为纯黑色；三个分量均为 255 则为纯白色）。对于计算机来说，真彩色图像是由数字组成的三维数组，因此，图 2.1 中的砖块图像在计算机中的表达即为 480

$\times 105 \times 3 = 151200$ 个数字。图像分类的任务即为对这数十万个数字进行数学运算，从而得到一个最大概率的类别标签，例如“裂缝”。



对于人来说，尤其是经验丰富的专业检测人员，对这样一个砖块图像进行分类显然是轻而易举之事。然而对于计算机来说，这样一个图像是以大量的数字来体现的，并且任意简单的几何变换（旋转、镜像）或亮度及对比度的细微调节，这些数字几乎发生了完全的变化，并且人眼中的同一类图像个体差异较大。例如裂缝的长度，宽度，走向不同，计算机读取到的也是看似完全没有任何联系的两组数，再加上例如光照、阴影等噪声因素的影响，使得计算机图像分类成为一项困难的工作。

在神经网络出现之前，已经具有很多基于机器学习方法的图像分类技术。例如 k 邻近分类器^[81]、SVM 分类器^[82]等等。所有的基于机器学习方法的图像分类技术均分为三个部分：输入、学习和评定。输入即为将一定数量的人工分类好的图像导入计算机，每张图像为其中一类，这些图像的组合称为“训练集”；学习即为使计算机基于训练集能够学到如何运算才能够正确地对图像进行分类，此步骤完成后即得到一个训练过的分类器；评定就是使用训练好的分类器对一些未参与训练的图像进行分类测试，将分类器的结果与真实类别对比，准确率越高说明该分类器效果越好。

k 临近分类器与神经网络没有任何联系，其大致的原理是将测试图像与训练图像进行以像素为单位的比较，并将像素差值绝对值累加，从而计算出两个图像间的“距离”， k 代表与测试图像距离最近的 k 个训练图像。此种方式由于需要将所有图像的每个像素进行计算差值和比较，造成测试效率很低，并且背景等一些无关因素会对整张图像像素的直接比较产生巨大的影响，准确率和适应性在实际任务中无法达到满意的效果。因此，在目前的图像分类任务中，一般不采用这种方法。

值得一提的是线形分类器^[83]，它是神经网络乃至深度学习领域的基础。其原理如下：

$$f(x_i, W, b) = Wx_i + b \quad (2.1)$$

这个函数被称为“评分函数”， x_i 代表图像， W 称为“权重”， b 称为“偏置项”。仍然以图 2.1 中的砖块为例，该砖块图像在计算机中表示为 151200 个数字，将这 151200 个数字组成一个 151200×1 的列向量。由于该函数的输出表示每个类别的评分，因此若将砖块分为 4 类（未损伤、裂缝、剥落、酥碱），那么输出的应是一个 4×1 的列向量。由线性代数的知识可知， W 应为一个 4×151200 的矩阵， b 为 4×1 的列向量。训练的过程即为更新 W 和 b 的过程，最终将测试图像输入训练好的线性分类器，即执行公式 2.1，得到各类的“评分”，评分最高类即为分类器所判定的类别，示例图如图 2.2 所示（假定将 151200 个数字简化为 6 个数字）。

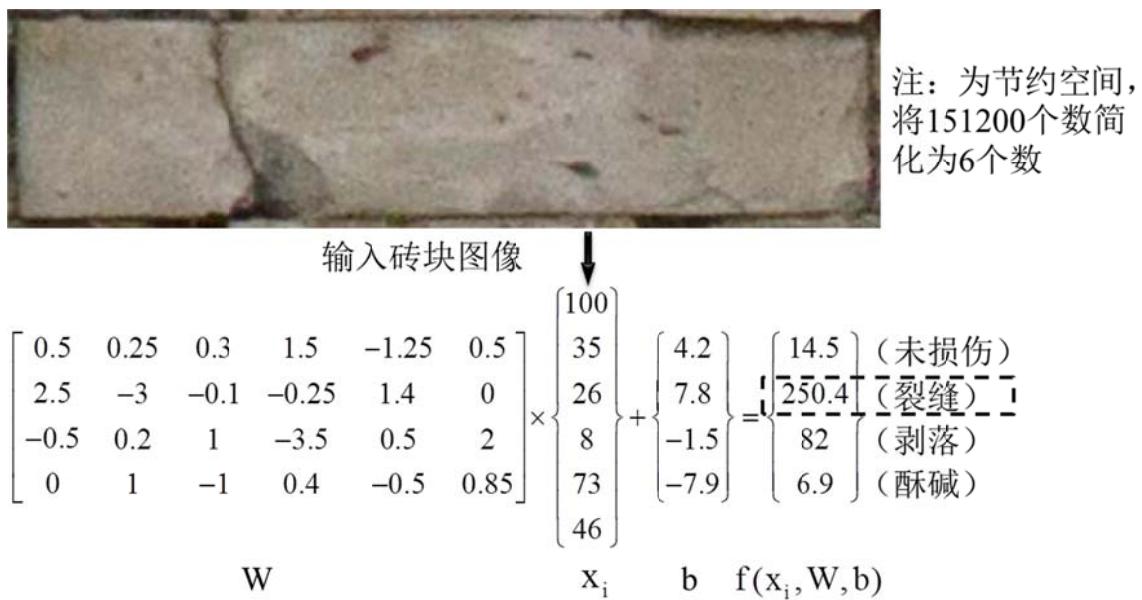


图 2.2 线性分类器示例

那么如何训练这样的一个分类器呢？实际上在训练的过程中，训练集中的图像也同样地执行公式 2.1。对于一张训练图像 x_i ，已经人工分类为 y_i ，也就是说， x_i 的真实类别为 y_i 。然而在训练时执行公式 2.1 的过程中，计算出 y_i 类别的评分值也许不是最大的，也许即使是最大的但是与其他类别的评分相差不多。图 2.2 例子计算出来的评分是一个很好的结果。因为该图像人工对其分类即为“裂缝”，同时可以看到，“裂缝”类别的评分远大于其它类别的评分。在这里定义“损失”的概念，“损失”代表了分类器的计算结果与真实类别之间的偏差，损失越小，分类器准确性越高，效果越好。

为了量化损失，需要借助于损失函数，最常见的为多元 SVM 损失^[84]和 Softmax 损失^[85]。多元 SVM 的原理为要求分类器计算得到的真实类别一类的评分必须高出其他类别一个边界值 Δ 。对于第 i 个样本，多元 SVM 定义的损失函数如下：

$$L_i = \sum_{k \neq y_i} \max(0, l_k - l_{y_i} + \Delta) \quad (2.2)$$

在这里 l 为类别评分数， k 为类别编号，可取 1 至类别总数， y_i 第 i 个样本真实类别编号， Δ 是边界值，是一个超参数^[86]。超参数是机器学习领域中的一个概念，简单来说超参数即使参数的参数。它们与例如包括权重和偏置项在内的学习参数不同，超参数一般为人工定义的配置参数。这些参数的设定大多没有理论基础，根据经验和根据训练效果不断地调整来得到最佳的超参数。仍然以图 2.2 为例， $k=1, 2, 3, 4$ ； $y_i=2$ ，这里令 Δ 取 10，那么该样本的损失函数 L_i 即为 $\max(0, 14.5 - 250.4 + 10) + \max(0, 82 - 250.4 + 10) + \max(0, 6.9 - 250.4 + 10) = 0$ 。

训练的过程就是调节 W 和 b 使得所有 L_i 尽可能小，甚至为 0。然而在实际的训练过程中，可能会有很多 W 和 b 满足这一条件，这时需要引入正则化惩罚^[87]。正则化惩罚是为了使分类器中学习参数的最小化和平均化，这样做的原因在于防止过拟合现象^[88]。因为学习参数对应着样本的某一维度特征，而为了具有普适性，分类器不可以特别关注某一维度。也就是说，为了具备较强的泛化能力，不能让任何一个维度特征过大影响着整体分类评价。一般来说仅对 W 进行正则化惩罚，因为 b 并不参与输入 x 的分类强度影响，对该项的正则化效果可以忽略不计。最常用的正则化惩罚是采用 L^2 范式：

$$R(W) = \sum_m \sum_n W_{m,n}^2 \quad (2.3)$$

因此，损失函数共包含两项：一是分类损失，二是正则化损失，完整的多元 SVM 损失函数如下：

$$L = \frac{1}{N} \sum_i L_i + \lambda R(W) \quad (2.4)$$

其中， N 为训练集的样本总数， λ 为超参数，需通过交叉验证的方式获得最佳值。

Softmax 损失与 SVM 损失有些不同，SVM 是将公式 2.1 的输出值直接作为类别评分值，而 Softmax 将原始的评分做归一化处理，将每一类的评分看做分类器认为该类是正确分类的概率。Softmax 分类器依然执行公式 2.1，但是将输出的每个类别的分量看作该类别未经归一化的对数概率。每个样本的 Softmax 损失表现为交叉熵损失^[89]，算式如下：

$$L_i = -\ln\left(\frac{e^{l_{y_i}}}{\sum_k e^{l_k}}\right) \quad (2.5)$$

其中自然对数括号里面的部分称为 Softmax 函数。前文提到 Softmax 将公式 2.1 的输出看作是未经归一化的对数概率，那么执行 Softmax 函数之后便完成了对数概率的转换以及归一化处理。这种处理方式在执行训练的过程中实际上是在求真实类别的负对数概率的最小值，类似于最大似然法的思想。

Softmax 损失函数的完整形式与 SVM 相同，同样包含分类损失与正则化损失。分类损失亦为整个数据集分类损失的平均值，只是 L_i 按照公式 2.5 计算。

由公式 2.2 和 2.5 比较可知，若不考虑正则项，对于 SVM，评分只要满足边界值 Δ ，损失就会判定为 0。满足了边界值，对于分数差异的大或者小，对损失并无影响。而对于 Softmax 来说，分数差异大的损失值必会小于分数差异小的。若能够选择好 Δ 的取值，SVM 对于特定分类任务会具有更好的效果，而 Softmax 则更加具有普适性。

训练的任务就是将损失值降到最低，显然沿着损失的梯度负方向调节 W 和 b 的效率是最高的。因此，将损失函数对 W 和 b 中的各个参数求偏导，然后以一定的步长调节参数，这个步长称为学习率。学习率同样是一个超参数，其选取对于训练效果具有重大的影响：若设置过低，训练会很慢；若设置过高，同样会造成收敛很慢的效果，且可能无法达到最小值（在最小值附近震荡），其原理可以由一个二元函数的最优化图像直观表现出来，如图 2.3 所示。

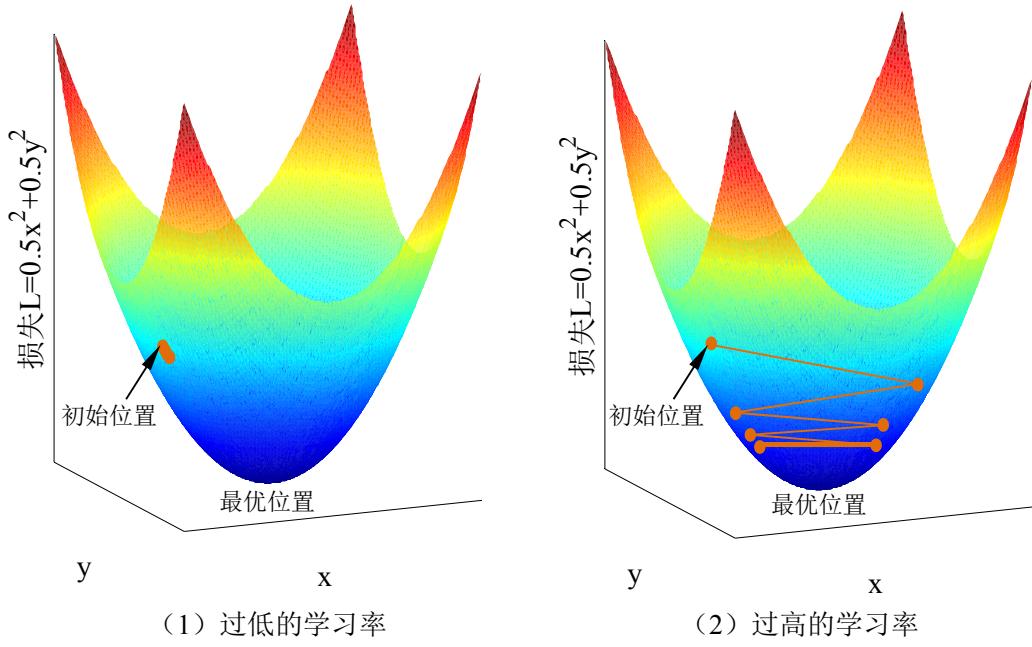


图 2.3 二元函数最优化图像示例

2.2.2 神经网络

线形分类器是人工神经网络的基础，事实上可以看做是没有激活函数的单层神经网络，例如将砖块是否损伤的分类映射到二维平面上，那么其达到的分类效果可由图 2.4 (1) 形象地表达出来。

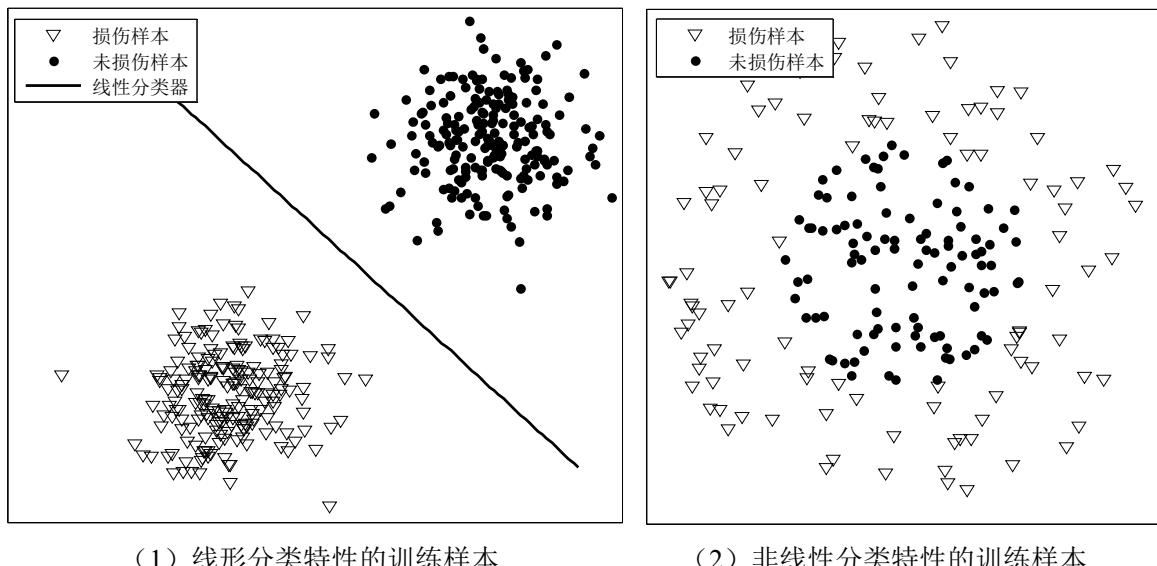


图 2.4 样本分类二维映射示例

实际情况中大多数的分类任务都不可能如图 2.4 (1) 那样简单, 而当样本体现出如图 2.4 (2) 所示的非线性特征时, 线形分类器便无法完成分类任务, 因此, 需要在分类器中加入非线性因素, 这就是激活函数的作用。

在机器学习领域中常见的激活函数有 Sigmoid 函数、tanh 函数和修正线性 (ReLU) 函数^[90], 如图 2.5 所示。除此之外, 还有近几年提出的 Leaky ReLU 函数^[91]、PReLU 函数^[92]和 RReLU 函数^[93]等等。

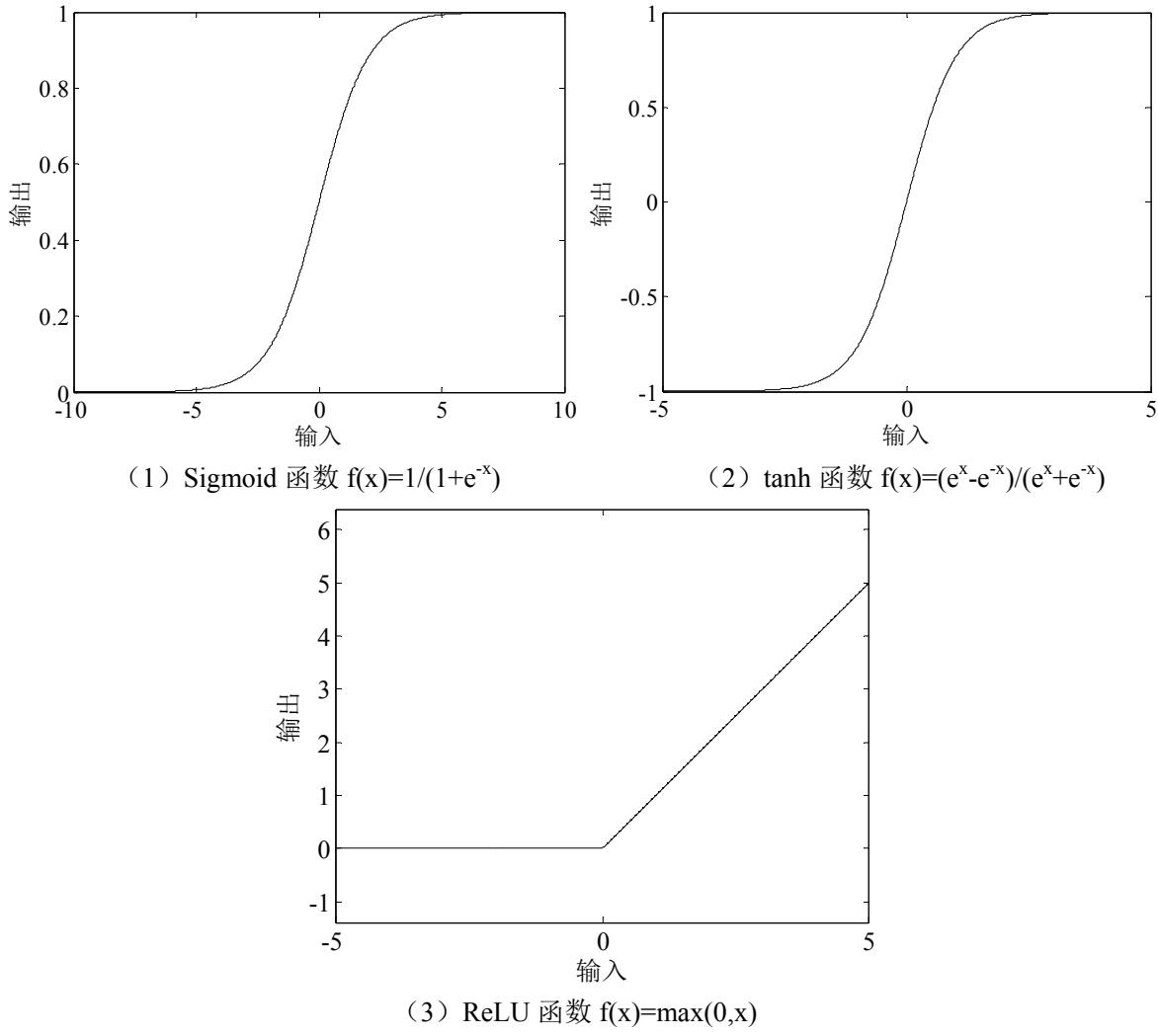


图 2.5 常见的激活函数

目前的深度学习领域使用最普遍的激活函数是 ReLU, 原因之一就是计算简单, 相比于 Sigmoid 和 tanh, ReLU 只需一个阈值计算, 很大程度上能够节约资源, 提高效率。同时, Sigmoid 和 tanh 很容易出现饱和现象, 即由图 2.5 (1) 和 (2) 可以清楚地看出, 当输入值超过原点的一定区域范围时, 梯度几乎为 0, 这在下文即将提到的反向传播过

程中将会导致网络近乎停止学习，是一个致命的缺陷。而 ReLU 函数由于其非饱和性，不存在这样的情况。然而由图 2.5 (3) 可知，ReLU 的缺陷在于训练的过程中，激活单元有时会很“脆弱”，即该单元的梯度将会不可逆转地永久变为 0，但是这种情况可以通过合理设置学习率来进行避免^[90]。在 Alex Krizhevsky 那篇关于 AlexNet 的论文中，实验发现使用 ReLU 的收敛速度高于 Sigmoid 和 tanh 数倍^[70]。

因此，通过激活函数，非线性的分类问题也得以解决。线性分类器通过激活函数，便构成了神经网络中的基本单元——神经元。若以 ReLU 函数作为激活函数（因为 ReLU 的优越性，本文后文若未特殊指明，激活函数均默认为 ReLU 函数），那么神经元模型表示如下：

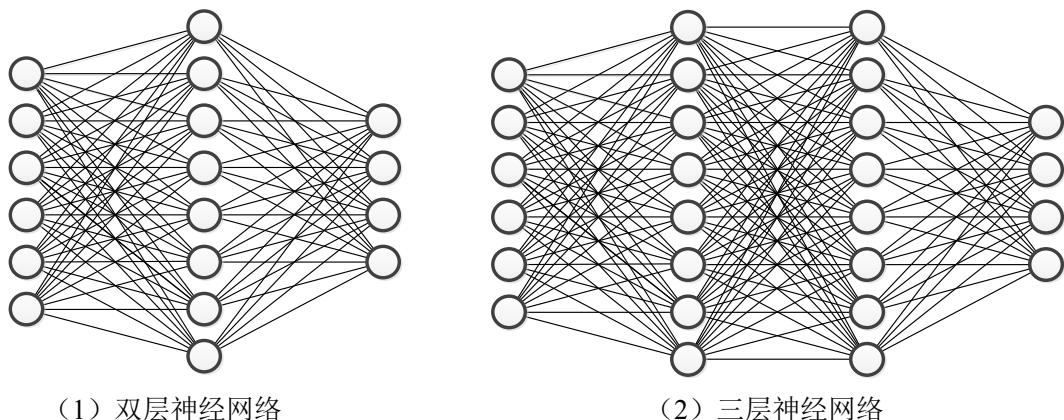
$$y_j = \max(0, \sum_{i=1}^n w_{ij} \times x_i + b_j) \quad (2.6)$$

其中 y_j 表示第 j 个神经元， w_{ij} 是权重 W 中的第 (i, j) 个元素， x_i 是输入向量的第 i 个元素， b_j 是偏置项 b 中的第 j 元素。

双层的神经网络实际上就是将神经元组合成一个列向量作为输入，在外面再加一个线性分类器。需要注意的是，由于最后的输出为类别评分或概率，一般最后一层不使用激活函数。双层神经网络模型表示如下：

$$y = W_2[\max(0, W_1 x + b_1)] + b_2 \quad (2.7)$$

其中， W_1 、 W_2 表示为第一层、第二层的权重， b_1 、 b_2 同理。更多层的神经网络即再输入激活函数并在外面添加线性分类器，中间的层叫做隐层。由矩阵乘法可知，隐层的神经元个数是超参数，其决定着该层的权重和偏置项的元素个数，以图 2.2 中的砖块分类为例，假设中间的隐层神经元均为 8，那么其图形化的结构如图 2.6 所示。



注：每个神经网络最左侧叫做输入层，最右侧叫做输出层，中间叫做隐层，计算层数时不计输入层

图 2.6 神经网络图形化结构

由图 2.6 可知, 在神经网络中层内的神经元互不连接, 而层间的神经元是全连接的, 这是最一般的神经网络结构, 这样的层叫做全连接层, 层间的全连接即通过权重 W 和偏置项实现。例如图 2.6(2), 该三层神经网络共有 $8+8+4=20$ 个神经元(输入层不计), $6 \times 8 + 8 \times 8 + 8 \times 4 = 144$ 个权重参数, $8+8+4=20$ 个偏置参数, 共 $144+20=164$ 需要学习的参数。在目前的深度学习领域, 一般的网络架构都具有上百万个神经元以及上亿的学习参数。

事实上, 早在 1989 年, Cybenko 的研究中已经证明, 只要选择合理的激活函数, 双层的神经网络分类器可以拟合成任何连续的函数^[94]。然而, 在神经网络发展的数十年中, 依然倾向于将网络越做越深。这是因为网络越深, 函数拟合的效果越好, 越平滑, 并且相对来说深的网络学习参数更加容易。

1986 年, Rumelhart 等学者提出的误差反向传播法^[65]使得神经网络计算效率大大提高, 使用此方法训练的神经网络称为 BP 神经网络, 是应用最广泛的神经网络之一。BP 神经网络的基本原理是, 输入的训练数据正向通过网络, 经过计算得到损失; 之后将损失通过反向逐层求梯度的方式调节学习参数。之后再次正向通过网络计算损失和反向求梯度, 如此循环, 直至网络收敛。

因此, BP 神经网络的训练即由正向传播和反向传播组成。正向传播的原理较为简单, 其实就是将输入数据通过神经网络得到当前参数下的损失值, 在这里说明一下反向传播。反向传播利用了梯度计算中的链式法则, 以一个双层的共 4 个神经元的神经网络中第一层某一权重参数 w_1 为例说明反向传播, 如图 2.7 所示。

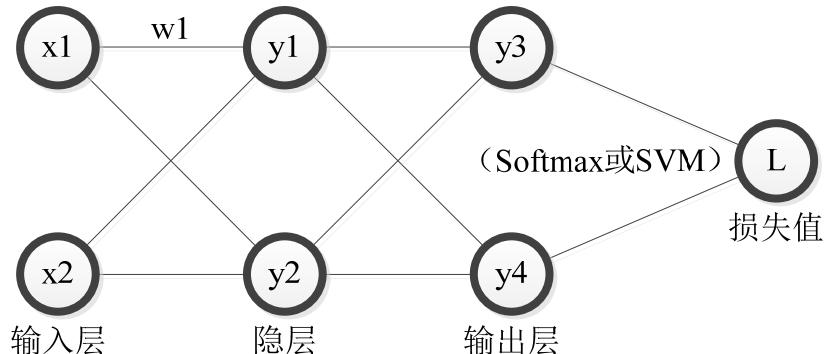


图 2.7 反向传播示例

对于 w_1 的反向传播如下式计算:

$$\frac{\partial L}{\partial w_1} = \left(\frac{\partial L}{\partial y_3} \cdot \frac{\partial y_3}{\partial y_1} + \frac{\partial L}{\partial y_4} \cdot \frac{\partial y_4}{\partial y_1} \right) \cdot \frac{\partial y_1}{\partial w_1} \quad (2.8)$$

若学习率为 1，则此次反向传播 w_1 的更新如下（更新后为 w_1' ）：

$$w_1' = w_1 - 1 \cdot \frac{\partial L}{\partial w_1} \quad (2.9)$$

2.2.3 卷积神经网络

一般来说，尤其对于图像分类任务，网络越深，效果越好。然而以全连接层为主的 BP 神经网络并不能做得很深。再次以图 2.1 中的砖块图像为例，假如使用隐层神经元个数为 100 的双层神经网络进行训练，那么需要学习的参数共有 $480 \times 105 \times 3 \times 100 + 100 + 100 \times 4 + 4 = 15120504$ 个，如果增加中间的隐层数量，再加上更大尺寸的图像，参数数目很容易达到上百亿个。巨大的参数数量本身对于计算机的运算速度来说就是一个极大的障碍，再加上海量的参数会很快造成过拟合，一般的 BP 神经网络能够做到 4 层就已经很深了。

卷积神经网络之所以能够做得很深，最主要在于它拥有一般 BP 神经网络所不具备的两个可以很大程度上减少参数数目的特征，即局部连接和参数共享^[63]。由于本文的研究内容是基于视觉的损伤识别，因此只讨论用于图像分类的卷积神经网络，这也是目前研究应用最为广泛，发展速度最快的卷积神经网络。

不同于线性分类器与一般的 BP 神经网络，卷积神经网络对于输入图像的处理方式并不是将其像素值排成一维的向量，而是直接将其看做一个三维的数组。例如图 2.1 中的砖块，卷积神经网络直接将其看做是一个宽度为 480，高度为 105，深度为 3 的三维数组。宽度和高度代表了砖块图像的实际宽度和高度的像素值，深度代表每个像素三原色的通道值。卷积神经网络对输入的数据在宽度和高度的方向上不断进行降维，但深度有时会增加，到最后的输出层减至和类别一样的数量。对于本文的分类数目为 4，那么最终通过卷积神经网络时，输出层将具有 $1 \times 1 \times 4$ 的维度，其图形化架构如图 2.8 所示。

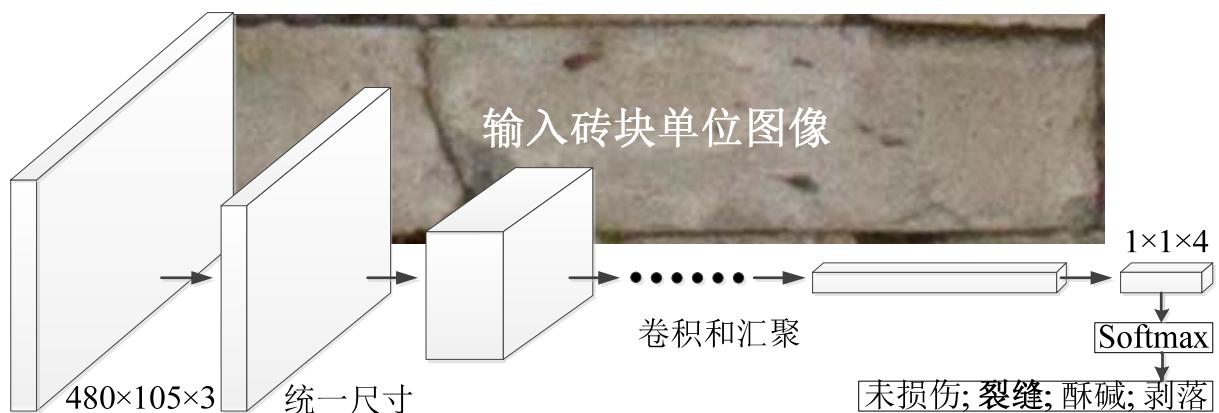


图 2.8 卷积神经网络图形化架构

相比于一般的 BP 神经网络，卷积神经网络特有的结构层主要包括卷积层和汇聚层两种。这两种结构层加上全连接层、激活函数层、损失函数层以及一些包括局部响应归一化层和随机失活层等在内的附加层，构成了一个完整的深度卷积神经网络。

(1) 卷积层

每个卷积层是一个三维的数据结构，此结构由若干二维平面组成，每个平面又是由大量的神经元构成。每个神经元对应一个偏置项，这与一般的 BP 神经网络并没有什么不同。不一样的地方在于权重项，一般的 BP 神经网络以全连接的方式传递信息，因此权重项是一个尺寸为前一层和该层神经元乘积的二维矩阵。而卷积层不同，其权重项称为卷积核，也叫作滤波器，一般宽度和高度方向尺寸相等且很小。相比于前一层例如图 2.1 砖块 480×105 的平面尺寸，卷积核一般具有 3×3 、 5×5 、 7×7 这样很小的平面尺寸，是一个超参数。然而卷积核的深度方向必须与前一层的深度一致，例如前一层是 $480 \times 105 \times 3$ 的砖块数据输入层，紧接着的这一层卷积层采用边长为 5 的卷积核，那么这一层卷积核的尺寸即为 $5 \times 5 \times 3$ 。卷积核以一定的步长在前一层数据的宽度和高度方向进行滑动，其每停留在一个位置，便和前一层于该位置的数据进行内积操作，即对应位置的元素相乘再累加，最后加上偏置项，得到的结果即为该层的一个神经元，这一过程便称为“卷积”。由于卷积核只在前一层宽度和高度的方向上滑动，而深度方向的神经元又不能够随意丢弃，因此卷积核深度的尺寸必须同前一层一致，即在内积的过程中深度方向的神经元全部参与卷积。

为了使卷积核在滑动的过程中能够以不变的步长整齐扫描前一层的每一个神经元，有时需要在卷积操作之前对前一层的边缘添充一定宽度的“零”神经元，这样做不仅可以解决滑动过程中的步长设置和整齐对称问题，同时可以控制这一层神经元宽度和高度方向的尺寸。同时，零填充并不会对图像分类造成任何的影响，因为在卷积的过程中，卷积核中任何参数与零相乘结果均为零，不会对累加的结果造成影响，因此输出神经元的结果与是否零填充及零填充的尺寸完全无关。

一般来说，为操作方便，卷积层的前一层一般都已经处理为高度和宽度相等的尺寸。若其边长为 L ，卷积核的边长为 C （超参数），零填充的尺寸为 Z （超参数），卷积核的步长为 S （超参数），那么卷积后的输出，即本层的边长 L' ，由数学关系可得：

$$L' = \frac{L - C + 2Z}{S} + 1 \quad (2.10)$$

卷积核滑过平面之后，会得到一个尺寸为由公式 2.10 计算得到的二维数据。若对该二维数据进行可视化操作，则会得到一个激活图，激活图上体现了卷积核在前一层数据

上不同位置的反应情况。简单地说，若其识别了前一层的某些特征，卷积之后的对应区域便会“激活”，其视觉上的体现可能斑点、蜂窝状的图案等等。

需要说明的是，每一层的卷积核一般有多个，这些卷积核同时在前一层的数据上进行滑动，每一个卷积核滑动之后形成一个二维平面， N 个卷积核经过卷积操作之后便形成了深度为 N 的三维数据，即本层神经元的集合。因为需要保证二维平面能够整齐叠加，所以每个卷积核的尺寸是固定的。一般来说，更多的卷积核识别效果更好，但是过多的卷积核又会造成过拟合问题，因此这又是一个超参数，需要通过不断地实践获得其最佳取值。直观上理解，每个卷积核识别的是前一层神经元的某一特征，卷积核越多，识别的特征就会越多，然而特征识别过多又难免会造成过拟合现象。大量的卷积核使得卷积之后的输出深度很深，这也是卷积神经网络中结构层的深度方向有时会增加的原因。

上文提到的“局部连接”和“参数共享”两个卷积神经网络巨大的优势全部是在卷积层的卷积过程中得以体现的，下面将详细说明这两大优点。

① 局部连接

前文已经对卷积操作原理进行了详尽的解释，不难看出，卷积核停留在某一位置时，仅仅与该位置的神经元进行内积操作而输出一个新的神经元。也就是说，这个新的神经元实际上只与前一层的某个区域的神经元进行连接，而不是像一般的BP神经网络那样同前一层的所有神经元进行连接。这样做好处在于，通过局部连接，网络能够更好地识别出图像的某些局部特征，例如本文研究的损伤识别，砖块的裂缝特征只在图像的特定位置体现，局部连接能够更高效地提取这些特征。同时，与全连接相比，局部连接大大减少了计算机运算的复杂度，效率提高显著，两者对比如图2.9所示。

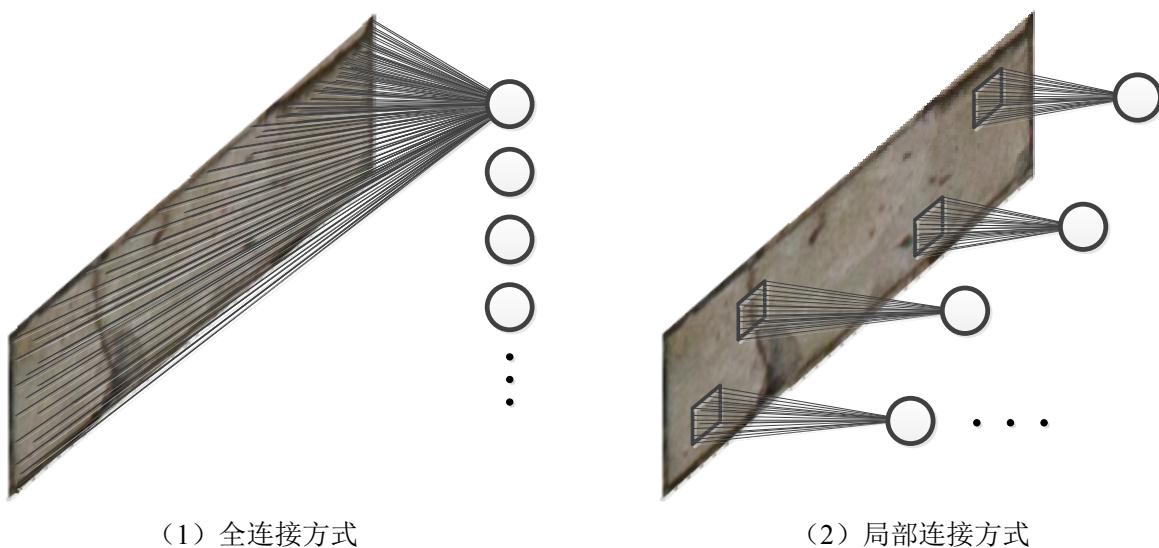


图 2.9 全连接方式与局部连接方式对比

② 参数共享

首先举例说明局部连接相对于全连接减少参数的数量：假设输入 256×256 像素分辨率的图像，接下来的卷积层仅使用一个尺寸为 3×3 的卷积核，步长设置为 2，零填充尺寸设置为 5，那么根据公式 2.10，经过卷积之后该层的边长尺寸为 $(256-3+5)/2+1=130$ 。由于只有一个卷积核，那么该层的神经元数目为 $130 \times 130 \times 1=16900$ 个。若按一般的 BP 神经网络使用全连接的方式，这一层需要学习的参数共 $(256 \times 256 \times 3+1) \times 16900=3322692100$ 。而使用局部连接的方式每个神经元仅需要 $3 \times 3+1=10$ 个参数，即该层共需要 $16900 \times 10=169000$ 个参数，直接减少了 4 个数量级。

然而例子中仅仅使用了一个卷积核，实际情况中每一层的卷积核是很多的，例如 AlexNet 的第一个卷积层就有 96 个卷积核^[70]，这样如果网络做得很深，参数数量依然巨大。为了继续优化参数数量，卷积神经网络采用同一卷积核参数共享的方式，即卷积核在滑动的过程中，其参数保持不变。也就是说，卷积之后形成的三维数据，其每一个宽度和高度方向的二维平面上的神经元，共用一套参数。因此，上面的那个例子由于只有一个卷积核，该层所有的神经元共享一套参数，故所需要学习的参数变成了 10 个。

卷积操作实际上就是在寻找并提取局部特征，那么从直观上理解，某一位置的分类特征在另外一个位置同样是适用的。例如某砖块有两条裂缝，那么用于判别第一条裂缝的特征同样适用于第二条，因此，参数共享不但可以大量减少参数数量从而实现并行训练计算，而且其本身设置也是合理高效的。

(2) 汇聚层

汇聚层亦称池化层，其主要功能为下采样，下采样的意思是将降低数据的特征维度。简单地说，通过汇聚层之后，数据在宽度和高度的方向上会大幅减小，而深度方向保持不变。下采样的主要原因还是为了减少参数避免过拟合现象，保留图像中的有效特征信息，并加快训练速度。

最常见的下采样方式为最大值区域采样，其他的方式包括平均值区域采样和 L^2 范式区域采样等等，但是目前绝大多数的卷积神经网络模型均使用最大值区域采样，并且在 Scherer 等学者的实践中已经证明，大多数情况下最大值区域采样要比其它采样方式训练效果好^[95]。因此，在本文中，若无特殊说明，汇聚层均采用最大值区域采样的方式。

因为汇聚层实际上并没有训练权重和偏置项的操作，因此该层不含有学习参数，只有两个超参数：汇聚区域尺寸 P 和步长 S 。所谓的下采样，就是对前一层的每一个二维数据平面使用 $P \times P$ 尺寸的区域以步长 S 在宽度和高度方向上滑动，每次停留时取该区域的最大值输出，其它值舍弃，最终实现宽度和高度方向上的降维。假设前一层的边长为 L ，零填充的尺寸为 Z ，那么下采样之后的输出，即本层的边长 L' ，由数学关系可得：

$$L' = \frac{L - P}{S} + 1 \quad (2.11)$$

例如，前一层为 $8 \times 8 \times 1$ 的数据， $P=S=2$, $Z=0$ ，那么经过下采样之后将会输出 $4 \times 4 \times 1$ 的数据，如图 2.10 所示。

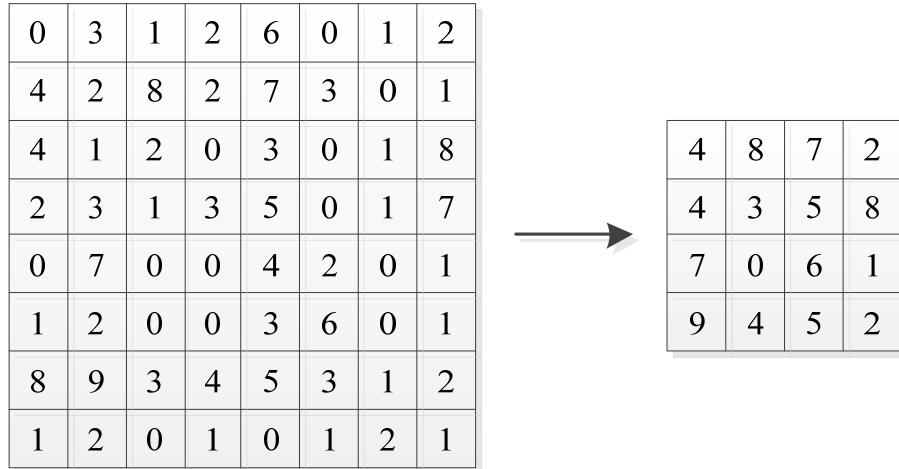


图 2.10 下采样示例

(3) 全连接层

卷积神经网络的全连接层与一般 BP 神经网络相同，主要应用于最后输出类别的评分。一般前一层的维度为 $1 \times 1 \times N$ ，经过全连接之后输出 $1 \times 1 \times N'$ ，权重为 $N' \times N$ 的二维矩阵，偏置项为 $1 \times 1 \times N'$ 的向量。

(4) 激活函数层

本文所有激活函数层均为 ReLU 层，前一层通过 ReLU 层后其每一个神经元 n 执行 $\max(0, n)$ 并输出，激活函数层没有学习参数和超参数。

(5) 损失函数层

卷积神经网络的损失函数层与一般的 BP 神经网络相同，本文全部采用 Softmax 层。

(6) 随机失活层

随机失活就是将输入该层的神经元以一定比率（超参数）置零，此操作可以大量减少参数数量，是一种效果很好的防止过拟合的手段。

(7) 局部响应归一化 (LRN) 层

LRN 层并不在每个网络模型中都出现，目前较深的网络模型一般都已经不再使用这一结构层^[73]。然而在关于 AlexNet 的论文中，实验表明，使用 LRN 层能够将错误率降低 1.4%^[70]。本文第四章的所有训练中均使用了 LRN 层。

LRN 层模仿生物神经元的“侧抑制”机制，对前一层的输入进行局部归一化处理，使局部变量的方差相同，加速训练。LRN 可以选择通道内归一化也可以选择通道间归一化，通道内归一化是以当前处理的神经元为中心，宽高方向 $n \times n$ 的二维区域进行处理；通道间归一化同样以该神经元为中心，深度方向为 n 的一维区域进行处理。输入的每一个神经元 x_i 执行下式，其中 n 、 α 、 β 均为超参数：

$$y_i = \frac{x_i}{\left(1 + \frac{\alpha \sum_{i=1}^n x_i^2}{n}\right)^\beta} \quad (2.12)$$

(8) 参数初始化

对于偏置项来说，一般均初始化为 0。对于权重，目前有多种不同的初始化方式，本文使用的有高斯初始化和 Xavier 初始化^[96]。高斯初始化原理简单，即给定均值和标准差，使初始权重参数形成高斯分布即可，然而此方法在网络较深的情况下收敛较慢。Xavier 原理较为复杂，利用了大量的统计学原理^[96]，主要解决了深层网络数值方差越来越小而造成梯度消失的问题，其对某一层初始权重 W_i 设置如下（ U 为均匀分布， N_{i-1} 为输入神经元维度， N_i 为输出神经元维度）：

$$W_i \sim U\left(-\sqrt{\frac{6}{N_{i-1} + N_i}}, \sqrt{\frac{6}{N_{i-1} + N_i}}\right) \quad (2.13)$$

(9) 参数更新

深度学习领域常用的参数更新方法包括随机梯度下降（SGD）+动量法（简称 SGD 法）^[97]、AdaDelta 法^[98]、自适应梯度（AdaGrad）法^[99]、Adam 法^[100]、Nesterov 法^[101]和 RMSprop 法^[102]。其中 SGD 法是应用最广泛且效果不错的方法，本文参数更新均采用这种方式。

SGD 法每次仅更新一部分训练样本的参数，这部分训练样本的数量称为一个 batch，更新一次一个 batch 的参数称为一个迭代，更新一次全部训练样本的参数称为一个 epoch。在训练的过程中，尤其是大量的迭代，学习率需要设置越来越低，否则可能永远达不到损失函数的最小值，就像图 2.3 (2) 一样。因此引入超参数 m ， m 被称为“动量”，一般取 0.9，其发挥的作用类似于物理学中的摩擦系数，在训练的过程中能够有效抑制学习率，达到“学习率退火”的目的^[97]。

以更新权重为例（偏置项同理），若 L 为损失函数， l 为学习率， D 为权重更新值（初值设为 0），那么权重更新如下：

$$D \leftarrow mD - l\nabla L(W) \quad (2.14)$$

$$\mathbf{W} \leftarrow \mathbf{W} + \mathbf{D} \quad (2.15)$$

2.3 网络架构及超参数配置

本文全部的样本训练均基于 8 层的 AlexNet^[70]和 22 层 GoogLeNet^[72]两种结构（层数的确定仅考虑具有学习参数的结构层，即卷积层和全连接层），AlexNet 是 ILSVRC-2012 的冠军模型，GoogLeNet 是 ILSVRC-2014 的冠军模型，两者都是非常具有代表性的经典卷积神经网络结构。

针对砖块损伤的特定图像分类问题，本文对 AlexNet 和 GoogLeNet 的某些超参数做了细微的调整，但仍然保留了基本的网络架构。因此，为尊重原作者，本文调整过的网络在下文中称为 AlexNet for HMDI 和 GoogLeNet for HMDI，HMDI 为“古建筑”、“砌体结构”、“损伤”和“识别”四个词语的英文首字母。

2.3.1 AlexNet for HMDI 架构及超参数配置

(1) 网络架构

AlexNet for HMDI 由卷积层、汇聚层、全连接层、ReLU 层、LRN 层、随机失活层和 Softmax 层共 7 种结构层组合而成，其网络架构如图 2.11 所示。

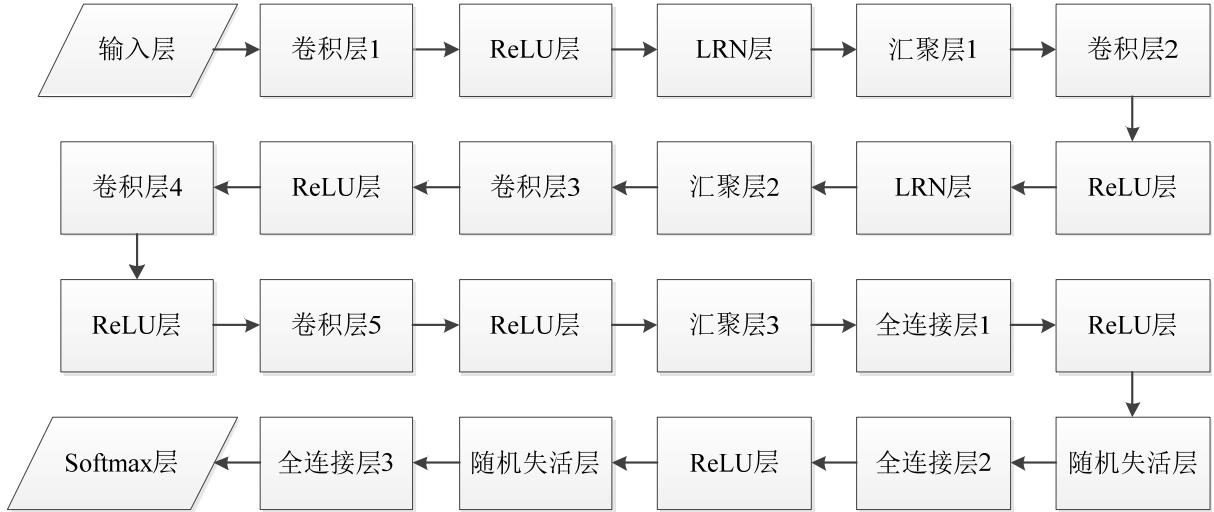


图 2.11 AlexNet for HMDI 网络架构

(2) 超参数配置

① 输入层

输入图像尺寸为 227×227 。

② LRN 层

$n=5$; $\alpha=0.0001$; $\beta=0.75$ 。

③ 随机失活层

随机失活率为 0.5。

④ 卷积层、汇聚层、全连接层

超参数见表 2.1。

表 2.1 AlexNet for HMDI 卷积层、汇聚层、全连接层超参数表

结构层	卷积核或汇聚区域					输出宽度	输出高度	输出深度
	边长	深度	步长	零填充	个数			
卷积层 1	11	3	4	0	96	55	55	96
汇聚层 1	3	-	2	0	-	27	27	96
卷积层 2	5	96	1	2	256	27	27	256
汇聚层 2	3	-	2	0	-	13	13	256
卷积层 3	3	256	1	1	384	13	13	384
卷积层 4	3	384	1	1	384	13	13	384
卷积层 5	3	384	1	1	256	13	13	256
汇聚层 3	3	-	2	0	-	6	6	256
全连接层 1	-	-	-	-	-	1	1	4096
全连接层 2	-	-	-	-	-	1	1	4096
全连接层 3	-	-	-	-	-	1	1	4

2.3.2 GoogLeNet for HMDI 架构及超参数配置

(1) 网络架构

GoogLeNet 与 AlexNet 的主要不同，除了深度的增加外，还提出了一种称为“ Inception ”的结构。在 Inception 结构中，将具有不同尺寸卷积核的卷积层并联，这种连接方式实现了网络同时对多种尺度范围特征的提取。LeNet 和 VGG-Net 等结构层全部串联的模型的一个缺点，就是对于图像的某一个尺度，只能用一种维度的卷积核进行处理。实际上图像可能具有不同尺度的特征，而这些特征有时无法简单地找到用于提取它们的卷积核的最优尺寸，因此采取不同尺寸的卷积核并联的方式能够更大程度上有效提取特征。为了防止参数太多造成过拟合，Inception 结构中在较大尺寸的卷积核前面以及汇聚层后面分别加入了 1×1 的卷积核进行降维，同时也起到了修正线性的作用。最后设置深度汇聚层，即将几个分支在深度方向进行组合并输出。

此外 GoogLeNet 在最后的全连接层前设置了一个全局平均汇聚层，再次有效降低了参数数量。同时，为了防止由网络过深造成的梯度消失现象，GoogLeNet 在网络中间增

加了两个辅助的 Softmax 层 (L1、L2) , 用于反向传播时增加前层的梯度。GoogLeNet for HMDI 网络架构如图 2.12 所示。

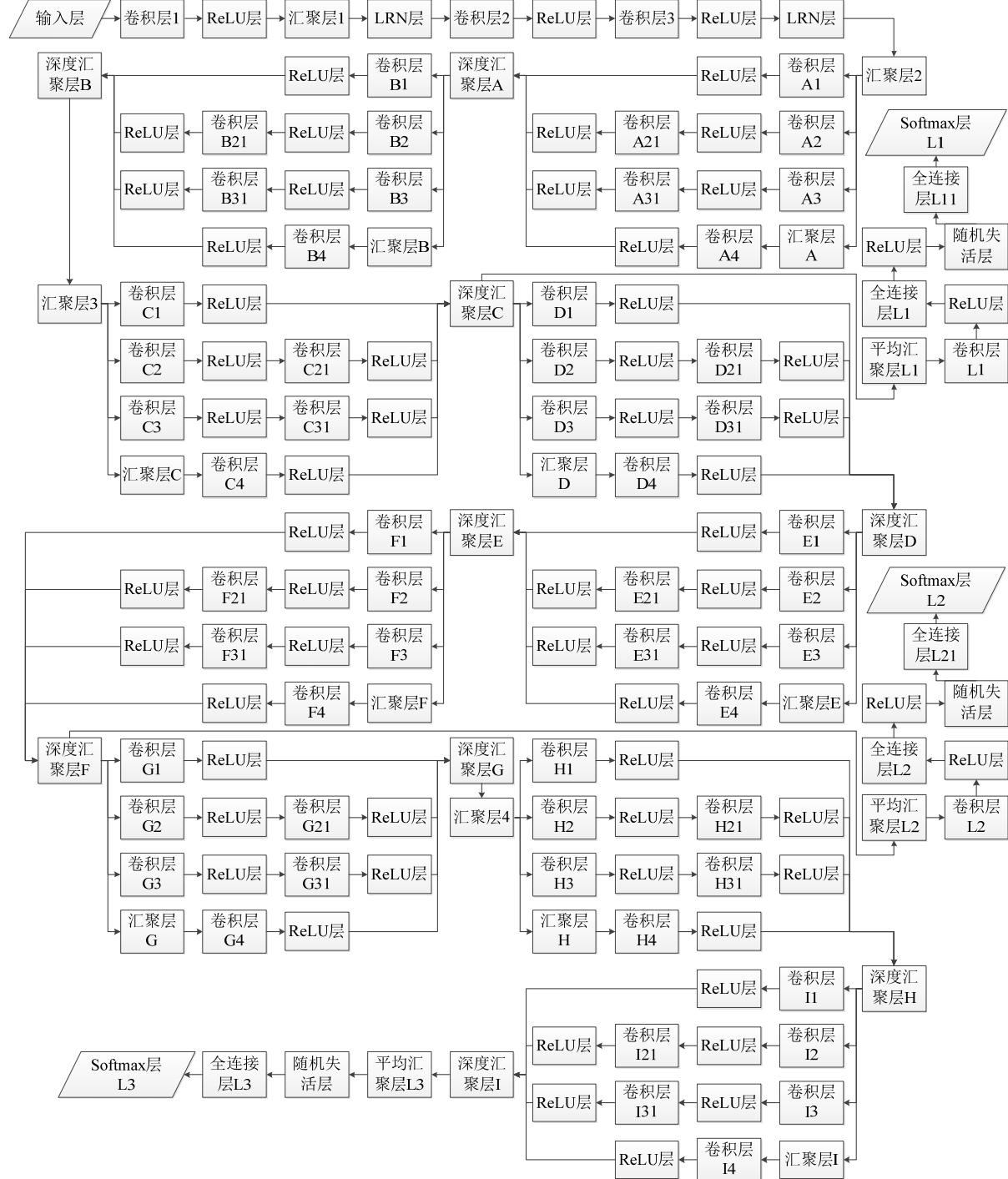


图 2.12 GoogLeNet for HMDI 网络架构

(2) 超参数配置

① 输入层

输入图像尺寸为 224×224 。

② LRN 层

$n=5$; $\alpha=0.0001$; $\beta=0.75$ 。

③ 随机失活层

Softmax L1 和 Softmax L2 前的随机失活层的失活率为 0.7, Softmax L3 前的随机失活层的失活率为 0.4。

④ 卷积层、汇聚层（包括深度汇聚和平均汇聚）、全连接层

超参数见表 2.2。

表 2.2 GoogLeNet for HMDI 卷积层、汇聚层、全连接层超参数表

结构层	卷积核或汇聚区域					输出宽度	输出高度	输出深度
	边长	深度	步长	零填充	个数			
卷积层 1	7	3	2	3	64	112	112	64
汇聚层 1	3	—	2	0	—	56	56	64
卷积层 2	1	64	1	0	64	56	56	64
卷积层 3	3	64	1	1	192	56	56	192
汇聚层 2	3	—	2	0	—	28	28	192
Inception A	卷积层 A1	1	192	1	0	64	28	64
	卷积层 A2	1	192	1	0	96	28	96
	卷积层 A21	3	96	1	1	128	28	128
	卷积层 A3	1	192	1	0	16	28	16
	卷积层 A31	5	16	1	2	32	28	32
	汇聚层 A	3	—	1	1	—	28	192
	卷积层 A4	1	192	1	0	32	28	32
	深度汇聚层 A	—	—	—	—	28	28	256
Inception B	卷积层 B1	1	256	1	0	128	28	128
	卷积层 B2	1	28	1	0	128	28	128
	卷积层 B21	3	256	3	1	192	28	192
	卷积层 B3	1	256	1	0	32	28	32
	卷积层 B31	5	32	1	2	96	28	96
	汇聚层 B	3	—	1	1	—	28	256
	卷积层 B4	1	256	1	0	64	28	64
	深度汇聚层 B	—	—	—	—	28	28	480
Inception C	汇聚层 3	3	—	2	0	—	14	480
	卷积层 C1	1	480	1	0	192	14	14
	卷积层 C2	1	480	1	0	96	14	14
	卷积层 C21	3	96	1	1	208	14	14
	卷积层 C3	1	480	1	0	16	14	14
	卷积层 C31	5	16	1	2	48	14	14
	汇聚层 C	3	—	1	1	—	14	14
	卷积层 C4	1	480	1	0	64	14	64

表 2.2 (续 1) GoogLeNet for HMDI 卷积层、汇聚层、全连接层超参数表

	深度汇聚层 C	-	-	-	-	14	14	512
L1 输出	汇聚层 L1	5	-	3	0	4	4	512
	卷积层 L1	1		1	0	128	4	128
	全连接层 L1	-	-	-	-	1	1	1024
	全连接层 L11	-	-	-	-	1	1	4
	卷积层 D1	1	512	1	0	160	14	160
Inception D	卷积层 D2	1	512	1	0	112	14	112
	卷积层 D21	3	112	1	1	224	14	224
	卷积层 D3	1	512	1	0	24	14	24
	卷积层 D31	5	24	1	2	64	14	64
	汇聚层 D	3	-	1	1	-	14	512
	卷积层 D4	1	512	1	0	64	14	64
	深度汇聚层 D	-	-	-	-	14	14	512
	卷积层 E1	1	512	1	0	128	14	128
Inception E	卷积层 E2	1	512	1	0	128	14	128
	卷积层 E21	3	125	1	1	256	14	256
	卷积层 E3	1	512	1	0	24	14	24
	卷积层 E31	5	24	1	2	64	14	64
	汇聚层 E	3	-	1	1	-	14	512
	卷积层 E4	1	512	1	0	64	14	64
	深度汇聚层 E	-	-	-	-	14	14	512
	卷积层 F1	1	512	1	0	112	14	112
Inception F	卷积层 F2	1	512	1	0	144	14	144
	卷积层 F21	3	144	1	1	288	14	288
	卷积层 F3	1	512	1	0	32	14	32
	卷积层 F31	5	32	1	2	64	14	64
	汇聚层 F	3	-	1	1	-	14	512
	卷积层 F4	1	512	1	0	64	14	64
	深度汇聚层 F	-	-	-	-	14	14	528
	汇聚层 L1	5	-	3	0	-	4	528
L2 输出	卷积层 L1	1		1	0	128	4	128
	全连接层 L1	-	-	-	-	1	1	1024
	全连接层 L11	-	-	-	-	1	1	4
	卷积层 G1	1	528	1	0	256	14	256
Inception G	卷积层 G2	1	528	1	0	160	14	160
	卷积层 G21	3	160	1	1	320	14	320
	卷积层 G3	1	528	1	0	32	14	32
	卷积层 G31	5	32	1	2	128	14	128
	汇聚层 G	3	-	1	1	-	14	528
	卷积层 G4	1	528	1	0	128	14	128
	深度汇聚层 G	-	-	-	-	14	14	832
	汇聚层 4	3	-	2	0	-	7	832
Inception H	卷积层 H1	1	832	1	0	256	7	256
	卷积层 H2	1	832	1	0	160	7	160
	卷积层 H21	3	160	1	1	320	7	320
	卷积层 H3	1	832	1	0	32	7	32
	卷积层 H31	5	32	1	2	128	7	128
	汇聚层 H	3	-	1	1	-	7	832

表 2.2 (续 2) GoogLeNet for HMDI 卷积层、汇聚层、全连接层超参数表

	卷积层 H4	1	832	1	0	128	7	7	128
	深度汇聚层 H	-	-	-	-	-	7	7	832
Inception I	卷积层 I1	1	832	1	0	384	7	7	384
	卷积层 I2	1	832	1	0	192	7	7	192
	卷积层 I21	3	192	1	1	384	7	7	384
	卷积层 I3	1	832	1	0	48	7	7	48
	卷积层 I31	5	48	1	2	128	7	7	128
	汇聚层 I	3	-	1	1	-	7	7	832
	卷积层 I4	1	832	1	0	128	7	7	128
	深度汇聚层 I	-	-	-	-	-	7	7	1024
L3 输出	汇聚层 L3	7	-	1	0	-	1	1	1024
	全连接层 L3	-	-	-	-	-	1	1	4

注：在卷积和汇聚的过程中，执行公式 2.10、2.11 时有时可能会得到非整数，默认处理方式为：卷积后输出尺寸向下取整，汇聚后输出尺寸向上取整。

2.4 本章小结

本章详细介绍了如何将深度学习（卷积神经网络）应用于古建筑砌体结构的表层损伤检测。卷积神经网络以线形分类器和一般的 BP 神经网络为基础，其近几年的飞速发展，再加上硬件环境的不断提高以及 GPU 加速训练的实现，使得更高效且精确的结构损伤识别技术的实现成为可能。从这一章的理论分析可知，对于单个砖块的损伤识别，仅仅需要训练样本、卷积神经网络模型的搭建以及计算机的自动训练，便可以构造出一个损伤识别分类器。将砖块图像输入该分类器，得到可靠的分类结果，便实现了“识别”这一目的。因此，下一章将结合实例，具体说明如何制作训练样本，以及如何搭建训练环境和网络模型，使计算机能够高效地进行样本训练，真正地从实际的角度实现这一技术。

3 建立数据集及准备训练环境

3.1 数据集的建立

训练一个分类器所需的所有样本的集合称为数据集，完整的数据集分为训练集、验证集和测试集三部分。训练集即作为样本输入，来调节网络的学习参数，从而实现分类器的初步建立；验证集则为检验分类器效果的数据，但并不是等到分类器完全调节好再进行检验，而是在训练的过程中以一定的时间间隔（迭代次数）使用验证集进行检验，即在训练的过程中通过验证集可以得到分类器的准确率变化，其本身不参与网络的迭代；测试集则是当分类器完全训练好（验证准确率达到满意值或已经进行了特定的迭代次数）之后，用分类器来执行分类任务工作的样本。对于本文的研究来说，训练集和验证集在外观和形式上并无差别，仅仅是用途上的区分。对于第四章基于滑动窗口的识别定位技术，训练集和验证集为人工分类过的单个砖块或砌块图像，测试集为含有多个砖块的砌体结构表层图像；对于第五章基于候选区域的目标检测技术，训练集和验证集为标记出损伤位置的部分砌体结构表层图像，测试集为未经标注的图像。

本文用于实验的数据集全部来自于中国北京故宫博物院古建部提供的故宫某一段像素分辨率为 57780×11400 的城墙的整体正投影图像。

对于第四章的研究，训练集的准备采取人工的方式从整体图像中截取出 5145 个砖块图像并进行损伤分类，经过与专业人员的讨论与鉴定，将这 5145 个砖块共分成 4 类：其中 1466 块砖定义为“未损伤”，1830 块砖定义为“剥落”，865 块砖定义为“裂缝”，984 块砖定义为“酥碱”，每种类别的示例如图 3.1 所示。



(1) 未损伤



(2) 剥落

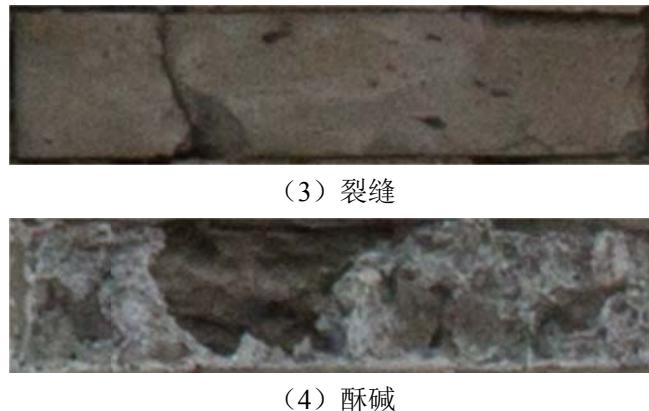


图 3.1 每类样本图像示例

验证集为另截取的 2000 块砖块图像，上述每一类别 500 块。测试集为远离训练集和验证集截取位置的 5 张像素分辨率为 1860×1260 的故宫城墙表层图像，每张图像包含 60 个完整的砖块。

由于深度学习需要庞大的数据量，仅仅 5000 多个训练样本对于四分类的任务显然是不够的。因此，本文对训练集进行数据增强操作。所谓的数据增强，就是对样本通过多种几何或亮度变换来成倍地扩充其数量。需要说明的是，数据增强并非单纯的变相复制。很多研究已经证明，其确实能够起到增加样本的作用^[103,104]，并且可以解决如本文类别数量不平衡的问题^[105]。事实上，AlexNet 中也运用了这个技巧^[70]。直观上解释，例如旋转图像，对于人眼来说类别是不会有任何变化的，损伤不会因为旋转就成了未损伤，然而对于计算机来说，旋转后是一个完全不同的三维矩阵，训练时也不可能获得完全相同的学习参数。本文对训练集进行了 90° 、 180° 和 270° 旋转、水平及竖直镜像、对比度变换以及它们的组合。对于类别数目较少的裂缝和酥碱样本进行相对较多的增强，对于其他类别进行相对较少的增强，最终扩充为一个每类 10000，共 40000 个样本的训练集。验证集、测试集不进行数据增强操作。

对于第五章的研究，由于带有一些初步的探索性质，本文的研究暂时仅截取 260 个像素分辨率为 760×700 的样本进行训练和测试，如图 3.2 所示。其中 200 个样本组成训练集，30 个样本组成验证集，30 个样本组成测试集。由于样本数目很少，暂时不对损伤进行进一步的分类，只单独研究损伤能否用第五章的方法成功地识别和定位。截取后对所有的样本进行标记处理，即通过人工的方式使用矩形方框将样本中所有损伤的砖块（无论何种损伤）全部标记出来，具体的标记方式在第五章中将会说明。



图 3.2 基于候选区域目标的损伤检测的数据样本

3.2 软硬件环境配置

为了保证分类器的训练工作能够顺利进行，须为其提供必要的高性能工作环境。近年来，随着对各领域对计算机运算能力的要求不断提高，NVIDIA 公司率先提出了使用 GPU 进行运算的技术^[106]。GPU 与中央处理器（CPU）最大的区别在于，CPU 核心数量很少，其主要功能是处理计算机的串行计算与优化任务；而 GPU 通常以更小的数以千计的核心组成，能够实现更为高效的并行计算与优化^[107]。用一个简单的比喻，CPU 计算类似于使用很少的几个顶级的士兵完成任务，而 GPU 计算类似于使用大量的普通的士兵去完成任务，对于具有大量相对简单工作的任务来说，GPU 计算的优势显而易见。而深度学习中的网络训练恰好具备这一特征，绝大部分工作只是十分简单的矩阵相乘相加，但是由于卷积神经网络深度的增加，工作的数量是巨大的。因此，GPU 在深度学习中扮演了一个几乎不可缺少的角色。Alex Krizhevsky 之所以能够在 ILSVRC2012 首次使用卷积神经网络获得冠军，主要原因之一正是 GPU 加速训练的实现^[70]。NVIDIA 公司在近几年中也在不断地升级其显示适配器，提高其 GPU 的运算能力，人工智能领域已经悄然进入了 GPU 计算时代^[108]。

实验部分使用的工作站相关硬件配置见表 3.1。

表 3.1 工作站相关硬件配置表

硬件	品牌及主要参数
计算机	Dell Precision Tower-7810
CPU	Intel Xeon E5-2630 v4 @2.2GHz (10 核心, 20 线程)
内存	32GB
GPU	NVIDIA Geforce GTX 1080 Ti (显存 11GB)
硬盘	ATA SK hynix SC300B SCSI Disk Device (512GB) ATA TOSHIBA DT01ACA2 SCSI Disk Device (2TB)

本文的网络训练及测试工作均在 Windows 环境下进行，并借助于 Caffe 深度学习框架（在 3.3 部分将会详述）。GPU 计算的实现需依靠 CUDA，CUDA 是 NVIDIA 公司推出的一个用于实现 GPU 运算的应用程序编写接口（API），也是一个通用并行计算的架构平台^[109]。本文应用的深度学习框架主要使用 C++语言，同时在训练和测试的过程中对结果的可视化处理时需要使用 MATLAB 和 Python 语言。Caffe 官方源代码在 GitHub 网站上取得^[110]，并在工作站上进行调整和编译。

实验部分使用的相关软件配置见表 3.2。

表 3.2 工作站相关软件配置表

项目	软件名称及版本
操作系统	Microsoft Windows 7 专业版 64 位 (6.1 7601)
C++	Microsoft Visual Studio Ultimate 2013 (12.0)
CUDA	NVIDIA CUDA Toolkit 8.0
cuDNN	NVIDIA cuDNN v5.1 for CUDA 8.0
MATLAB	MATLAB R2015b
Python	Python 2.7.13 (Anaconda2 4.3.1)

3.3 Caffe 深度学习框架的搭建

3.3.1 Caffe 简介

Caffe^[111]是由本科毕业于清华大学的贾扬清在加州大学伯克利分校攻读博士学位期间开发的一个深度学习框架，其完全基于 C++和 CUDA，并支持 MATLAB 和 Python 接口。不同于其它如 Torch、Theano、Tensorflow、Keras 和 MXNet 等框架^[112]，使用 Caffe 完全无需自己编写卷积神经网络的源程序，开发者将每个类型的结构层以及连接方式全部集成于各个 API 中，构建网络时仅需编写一个.prototxt 文件，将结构层类型以及超参数定义好即可。这种简介高效的特性，使得 Caffe 目前仍然是最流行的深度学习框架^[112]。

Caffe 包含四部分：二进制大对象（BLOB）、层、网络和求解器（Solver）。

(1) BLOB

BLOB 具有统一的内存接口，用于图像批处理以及更新学习参数，也就是说，BLOB 存储了卷积神经网络中的所有图像数据、权重、偏置以及激活值。Caffe 将这些数据以四维数组的方式存储于 BLOB 中，其中一维用于储存未经处理的原始数据，另外一维用于储存与数据相关的梯度及参数更新。因为 Caffe 即支持 GPU 训练也支持 CPU 训练，所以上面两个维度的数组均有 CPU 和 GPU 两个版本，故一共四个维度。

(2) 层

Caffe 中的层即为卷积神经网络中的结构层的实际体现，层会接收一个或多个 BLOB 的输入，Caffe 称这些 BLOB 为底端输入，之后会产生一个或多个 BLOB 输出，Caffe 称之为顶端输出。其原理为，层能够接收指向包含数据的底端输入的指针，也会接收指向顶端输出的指针。正向传播时，层最终将数据填在顶端的 BLOB 中；反向传播时，层能够实现梯度算法，它们能够接收指向包含梯度和激活值的顶端 BLOB 的指针，也会接收指向底部已经存有梯度的 BLOB 的指针。层是 Caffe 中具有非常好的结构性和条理性的一部分。

(3) 网络

“网络”更应该称为“连接”，其作用为将多个层连接在一起，实际上网络就是层的定向非循环流程图，保证卷积神经网络按照正确的顺序运算和更新参数，即正向传播和反向传播。在 Caffe 中，网络模型均为点对点的深度学习系统，一般的卷积神经网络起始于数据输入层，终止于损失评分层。

(4) Solver

Solver 的功能即为进入到网络中，通过正向传播和反向传播的方式使用数据来运行网络，更新网络中的学习参数，并可以建立中途的数据储存检查点。一般来说，网络训练是一个漫长的过程，根据网络深度、数据量以及工作站的硬件环境，训练时间通常在几个小时至数天不等。通过检查点，即使训练中断也无需重新开始，检查点会记录训练的信息且保存该点的训练模型，继续训练时直接将其调用即可。

Solver 是 Caffe 中一个较为抽象的部分，参数的不同更新规则由 Solver 中不同的子类来进行实现。例如本文 2.2.3 部分提到的 6 中参数更新方式，在 Caffe 的 Solver 中均可以实现，仅需将配置文件中的参数更新类别填写为想要的方式即可。

3.3.2 Caffe 的编译

配备好所有的硬件和软件环境之后，便可以进行 Caffe 的搭建了。首先从 Github 网站上下载 Windows 版本 Caffe 的官方源代码，解压文件夹后修改里面的配置文件

(CommonSettings.props.example) 并将文件名中的.example 去掉。本文修改的部分如图 3.3 所示。

```

7   <CpuOnlyBuild>false</CpuOnlyBuild>
8   <UseCuDNN>true</UseCuDNN>
9   <CudaVersion>8.0</CudaVersion>
10  <!-- NOTE: If Python support is enabled, PythonDir (below) needs to be
11  | set to the root of your Python installation. If your Python installation
12  | does not contain debug libraries, debug build will not work. -->
13  <PythonSupport>true</PythonSupport>
14  <!-- NOTE: If Matlab support is enabled, MatlabDir (below) needs to be
15  | set to the root of your Matlab installation. -->
16  <MatlabSupport>true</MatlabSupport>
17  <CudaDependencies></CudaDependencies>
```

图 3.3 Caffe 配置文件的修改

代码第 7 行（图 3.3 左侧的数字表示为代码的行数）表示 CPU 和 GPU 版本 Caffe 的切换，true 表示只编译 CPU 版本的 Caffe，false 表示 CPU 的 GPU 两种版本的 Caffe 均编译；第 8 行表示是否使用 cuDNN 加速，若选择“是”（true），同时要在代码的后半部分对应的位置填写其路径；第 9 行填写工作站中 CUDA 的版本；第 13 行和 16 行表示是否使用 Python 或 MATLAB 接口，若使用，同样需要在对应位置填写路径。

修改后，根据工作站的软件环境，仍应做一些细微的调整，例如增加环境变量，安装 OpenCV 等依赖库，修改 Visual Studio 中的项目属性等等。调整之后便可以打开 Caffe.sln 文件并对其中的项目进行编译，编译的顺序为：libcaffe→caffe→除 pycaffe 和 matcaffe 外的其他项目→pycaffe 和 matcaffe。初次的编译是一项十分繁琐的工作，有可能会遇到各种类型的错误，针对不同的问题采取对应有效的解决方式并重新编译，最终成功后将会在 Build 文件夹中生成 Caffe 在训练网络时将会调用到的所有文件。训练、测试或数据预处理时，只需编写对应的脚本即可高效地完成工作。

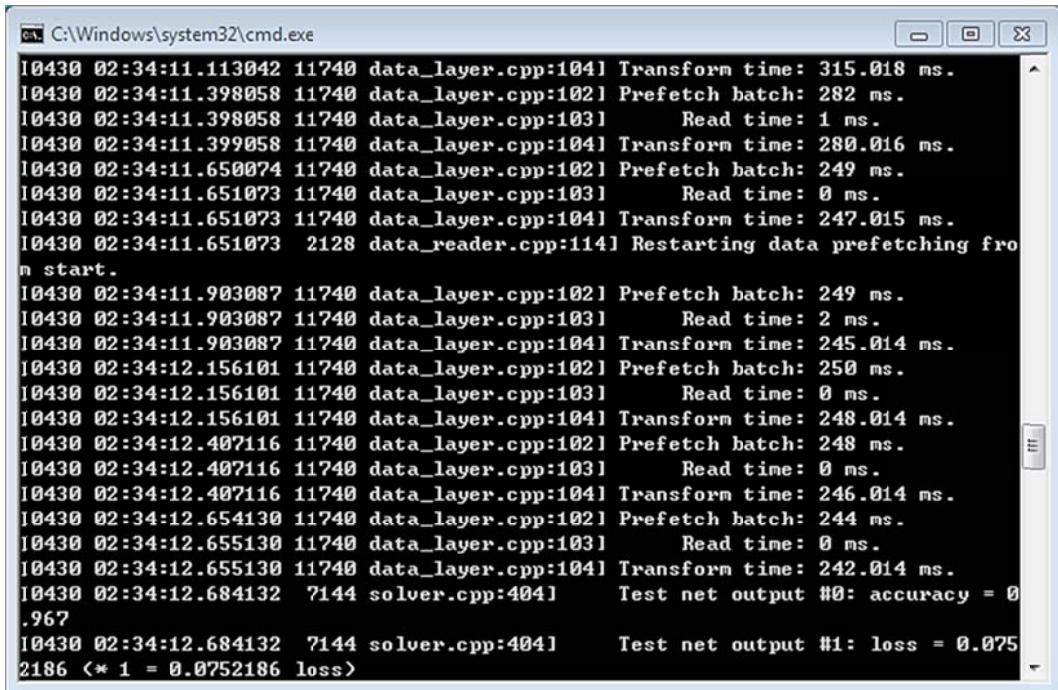
3.3.3 卷积神经网络在 Caffe 中的实现

使用 Caffe 训练网络时主要需要配置两个参数文件，即网络模型文件（train_val.prototxt）和求解器配置文件（solver.prototxt）。但是在配置这两个文件之前需要对训练集和验证集进行数据转换，即将输入的图像样本及人工分好的类别标签转化为数据库格式。Caffe 共支持三种格式的输入：LMDB、LEVELDB 和 HDF5。若转换为 LMDB 或 LEVELDB 格式，编写脚本调用如本文 3.3.2 部分所述编译出来的 convert_imageset.exe 文件即可；若转换为 HDF5 文件，则需单独编写程序完成工作，在此不赘述。

转换文件后即可开始训练的工作，然而一般情况下，为提高效率并达到更好的效果，训练之前需对数据进行预处理，最常见的预处理方式为零均值化，亦是本文所采取的方式。所谓的零均值化，就是将所有输入图像（训练集和验证集分开处理）的像素平均值计算出来，再用每个像素值减去该平均值得到新的图像代替原来的图像作为输入。这样做的好处在于可以降低由较大均值产生的梯度膨胀影响，并且若进行一些后续的处理工作，例如主成分分析，需要数据的均值为 0。需要说明的是，零均值化并没有改变图像的有效特征信息，因为其并没有改变像素之间的相对差值。例如将灰色调的砖块图像整体变换为白色调，通过人眼仍然能够判断出其是否损伤，正如计算机对图像的识别也并非根据其绝对像素值一样。

所有的数据预处理完成之后便开始进行参数文件与求解器文件的配置工作，文件中需要详细设定每一个超参数以及调用数据和储存数据的文件路径。

全部配置完成后，编写脚本即可开始网络训练，训练的过程中数据将会在命令控制台实时显示，如图 3.4 所示。为了能够更加清楚地了解训练和测试效果，还需借助 MATLAB 和 Python 对结果进行进一步的可视化处理，处理后的结果将在本文第四章结合实际的网络训练中进行说明。



```

C:\Windows\system32\cmd.exe
I0430 02:34:11.113042 11740 data_layer.cpp:104] Transform time: 315.018 ms.
I0430 02:34:11.398058 11740 data_layer.cpp:102] Prefetch batch: 282 ms.
I0430 02:34:11.398058 11740 data_layer.cpp:103] Read time: 1 ms.
I0430 02:34:11.399058 11740 data_layer.cpp:104] Transform time: 280.016 ms.
I0430 02:34:11.650074 11740 data_layer.cpp:102] Prefetch batch: 249 ms.
I0430 02:34:11.651073 11740 data_layer.cpp:103] Read time: 0 ms.
I0430 02:34:11.651073 11740 data_layer.cpp:104] Transform time: 247.015 ms.
I0430 02:34:11.651073 2128 data_reader.cpp:114] Restarting data prefetching from start.
I0430 02:34:11.903087 11740 data_layer.cpp:102] Prefetch batch: 249 ms.
I0430 02:34:11.903087 11740 data_layer.cpp:103] Read time: 2 ms.
I0430 02:34:11.903087 11740 data_layer.cpp:104] Transform time: 245.014 ms.
I0430 02:34:12.156101 11740 data_layer.cpp:102] Prefetch batch: 250 ms.
I0430 02:34:12.156101 11740 data_layer.cpp:103] Read time: 0 ms.
I0430 02:34:12.156101 11740 data_layer.cpp:104] Transform time: 248.014 ms.
I0430 02:34:12.407116 11740 data_layer.cpp:102] Prefetch batch: 248 ms.
I0430 02:34:12.407116 11740 data_layer.cpp:103] Read time: 0 ms.
I0430 02:34:12.407116 11740 data_layer.cpp:104] Transform time: 246.014 ms.
I0430 02:34:12.654130 11740 data_layer.cpp:102] Prefetch batch: 244 ms.
I0430 02:34:12.655130 11740 data_layer.cpp:103] Read time: 0 ms.
I0430 02:34:12.655130 11740 data_layer.cpp:104] Transform time: 242.014 ms.
I0430 02:34:12.684132 7144 solver.cpp:404] Test net output #0: accuracy = 0.967
I0430 02:34:12.684132 7144 solver.cpp:404] Test net output #1: loss = 0.075
2186 <* 1 = 0.0752186 loss>

```

图 3.4 训练过程在命令控制台中的实时显示

3.4 本章小结

本章详细介绍了数据集的建立以及训练环境的搭建和配置。通过人工和计算机结合的方式分割训练样本，并对其进行人工分类和标注，虽然工作较为繁琐，但是当数据库建立后，将具备很强的普适性。同时，选择 Caffe 这一深度学习框架实现本文的研究，具备如下优点：

- (1) 对于一般的前馈卷积神经网络，Caffe 可以很容易地实现
- (2) Caffe 可以对网络模型进行直接调整
- (3) 完全无需自己编写卷积神经网络的源程序，只需编写配置文件和脚本即可
- (4) 支持 MATLAB 和 Python 接口，功能强大

从下一章开始，将结合实际训练实验，并对比其它方式，详细说明本文检测方法的准确性、高效性和实用性。

4 基于滑动窗口算法的损伤识别与定位技术

古建筑砌体结构的表层损伤检测分为识别和定位两大部分。本章基于第二章的理论基础，应用第三章建立的数据集、配置好的训练工具和环境真正实施这一工作。通过一系列的实验，实现了对北京故宫城墙的表层损伤检测。

4.1 损伤识别技术

这一部分首先应用 AlexNet for HMDI 进行了较为简单的二分类实验，将裂缝、剥落和酥碱三种类别合为一类，即不区分损伤类别，并在实验中使用了不同数量的训练样本，用于研究训练集的大小对识别效果的影响。接下来进行区分损伤类别的四分类实验，同时在四分类的实验中将卷积神经网络深度的影响考虑在内，分别使用 AlexNet for HMDI 和 GoogLeNet for HMDI 两个模型作为不同的网络深度条件，在每个条件下使用不同数量的训练样本进行实验，从而得到四分类情况下网络深度和样本数量对于识别效果的影响。

本文的网络训练均采取 SGD 法，通过对卷积神经网络的初步调节测试，对深度相对较浅但学习参数较多的 AlexNet for HMDI 和深度相对较深但学习参数较少的 GoogLeNet for HMDI 采取两组不同的学习率超参数。

AlexNet for HMDI 的基础学习率设置为 0.001，学习率衰减系数设置为 0.1，衰减步长设置为 5000，其学习率随迭代次数变化如图 4.1 所示。

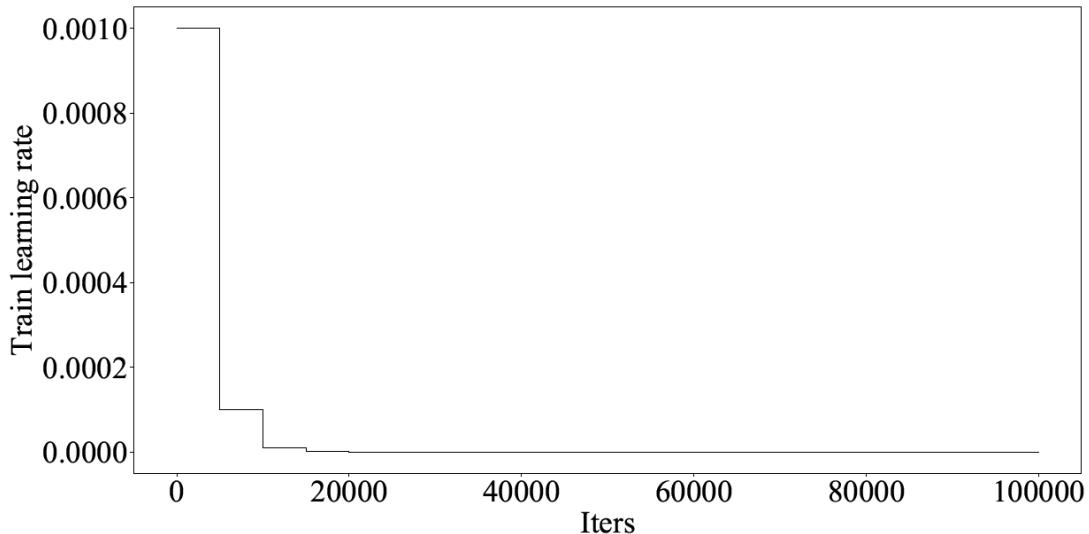


图 4.1 AlexNet for HMDI 学习率-迭代次数曲线

GoogLeNet for HMDI 的基础学习率设置为 0.001，学习率衰减系数设置为 0.96，衰减步长设置为 5000，其学习率随迭代次数变化如图 4.2 所示。

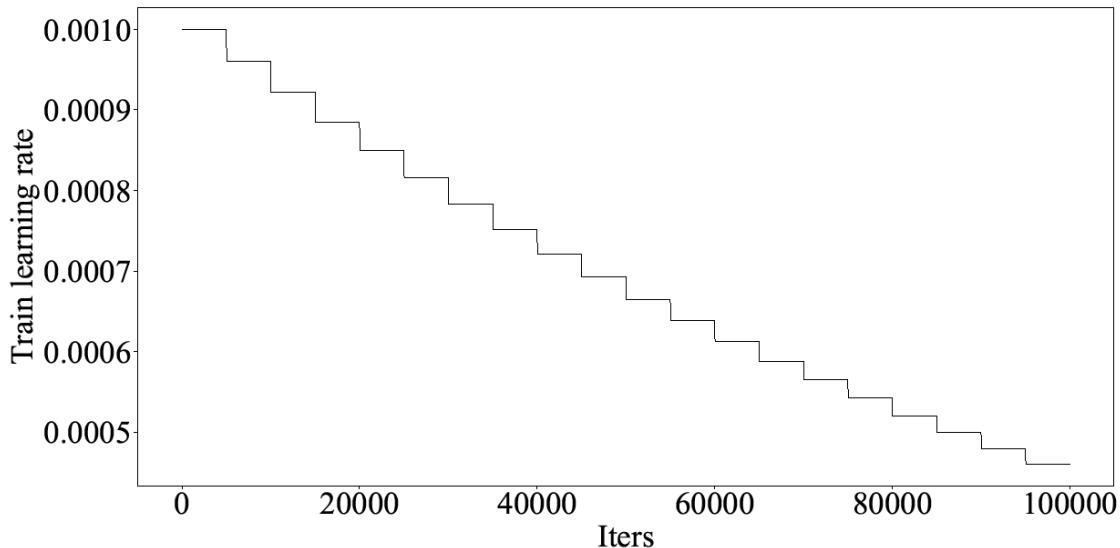


图 4.2 GoogLeNet for HMDI 学习率-迭代次数曲线

由于硬件条件的限制，所有的实验中，训练的 batch 设置为 64 个样本，验证的 batch 设置为 50 个样本。

4.1.1 二分类实验

(1) 基于少样本的二分类实验

从原始未经数据增强的 5145 个训练样本中提取 1000 个未损伤样本、333 个裂缝样本、333 个酥碱样本和 334 个剥落样本，将三类损伤合并为一类，这样，每一类训练样本均为 1000 个；从 2000 个验证样本中提取 100 个未损伤样本、33 个裂缝样本、33 个酥碱样本和 34 个剥落样本，同样将三类损伤合并，得到每类有 100 个验证样本的二分类验证集。

AlexNet for HMDI 要求输入图像像素分辨率为 227×227 ，因此需对样本图像进行缩放处理。需要说明的是，对样本的缩放并不会改变分类的特征，因为所有的砖块表层损伤都并非根据某一特征的绝对尺寸进行判定。也就是说，将砖块图像进行拉伸或缩放，对其分类几乎没有任何影响。对于此项预处理，Caffe 提供了一个很方便的接口，只需在图像转换为数据库格式时加入一行命令即可。

由于本实验分类数目为 2，故 AlexNet for HMDI 模型最后的全连接层神经元数目应改为 2。同时，本实验的验证集样本总数为 200 个，验证 batch 尺寸为 50，故每次验证

所计算的 batch 数目为 $200/50=4$ 。动量系数设置为 0.9，正则化系数设置为 0.0005，训练每迭代 20 次显示一次损失值，500 次进行一次验证，最大迭代次数为 100000。

网络训练耗时 9 小时 54 分 36 秒，实验结果如图 4.3 所示。

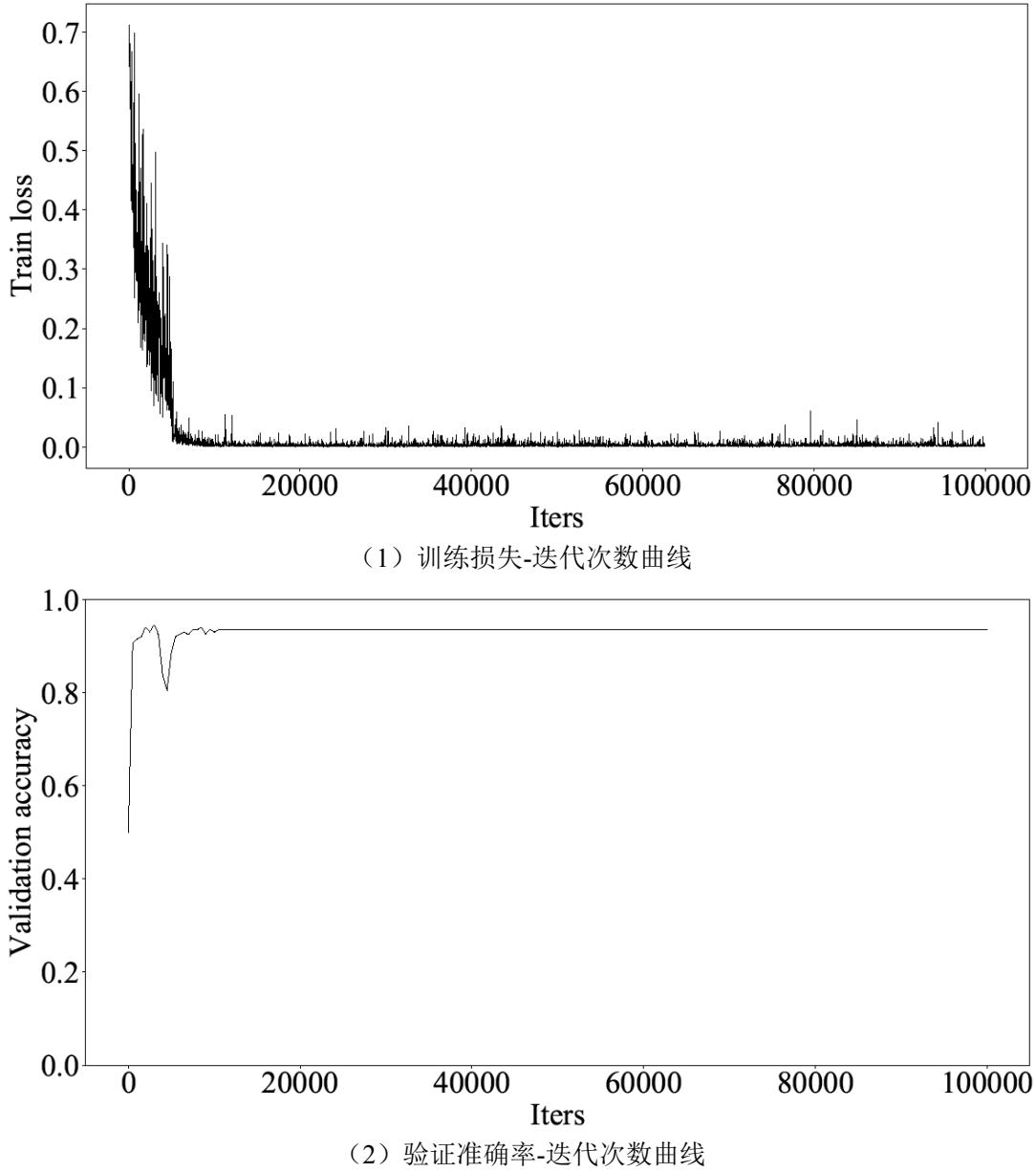


图 4.3 基于少样本的二分类实验结果

由图 4.3 中的曲线形状可知，训练损失值从 0.7 左右开始迅速下降，在 5000 次迭代时趋于平缓。验证准确率从初始的 50% 迅速上升，中间有少许下降和回升，在迭代 15000

次时已基本收敛，收敛较快一是由于分类数目较少，且未损伤与损伤特征区别较为明显，二是由于网络相对较浅，在后续的 GoogLeNet 实验中可知后者是主要原因。

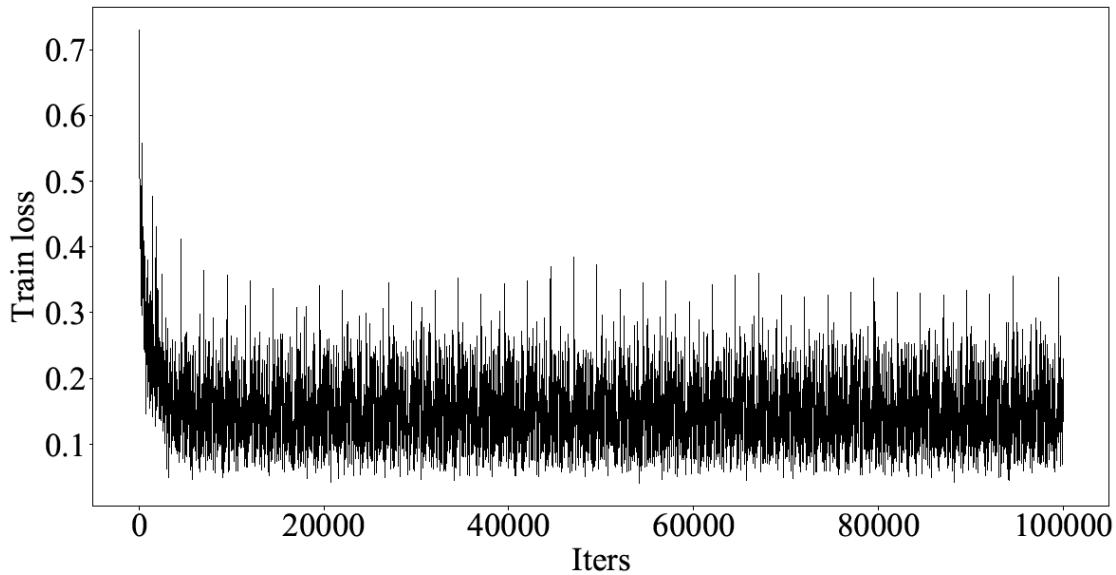
分类器最终的准确率稳定于 93.5%，对于二分类的任务来说，此准确率尚可，但是对于高标准的检测要求仍有一点点距离，因此接下来的实验中将使用更多的数据进行训练，提升分类器的准确率。

(2) 基于多样本的二分类实验

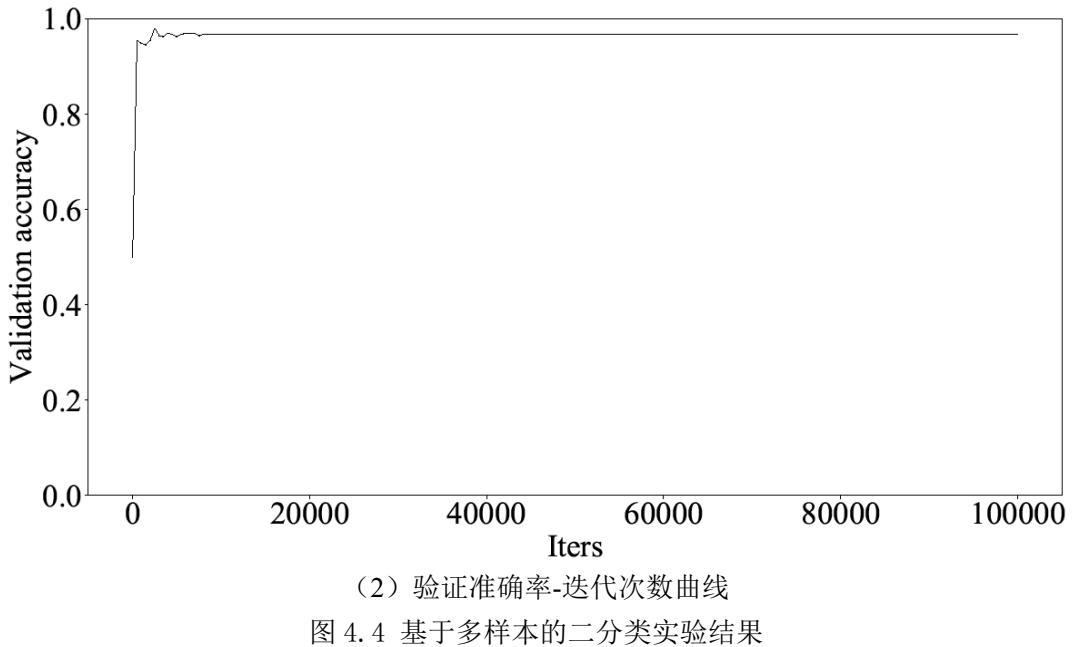
为了研究样本数量的增加能否提高二分类准确率，这一部分的训练集中使用数据增强的样本。将训练集全部的 10000 个未损伤样本取出，其它三类分别取出 3333 个裂缝样本、3333 个酥碱样本和 3334 个剥落样本，合并为一类，这样就形成了每类样本为 10000 个的二分类训练集，相比于前一个实验扩充了 10 倍的训练数据量。将验证集全部的 500 个未损伤样本取出，其它三类分别取出 166 个裂缝样本、167 个酥碱样本和 167 个剥落样本并合并，组成包含 1000 个样本的验证集。

由于验证集样本数的变化，求解器文件中的每次验证所计算的 batch 数目改为 $1000/50=20$ ，其余超参数保持不变。

训练耗时 9 小时 57 分 36 秒，实验结果如图 4.4 所示。



(1) 训练损失-迭代次数曲线



从实验结果可以看出，训练的损失下降后不是很稳定，在 0.15 附近波动明显，可能是样本数量较多，学习率下降不够快的原因。尽管如此，验证准确率却在大约 10000 次的迭代后稳定在 96.7% 不变，这对于损伤识别来说已经是一个很高的准确率。因此可以得出结论：样本数据量的增加对于分类器的损伤识别效果提高显著。这也是深度学习需要海量数据支持的原因，在网络结构不变的情况下，扩充数据是一种十分有效的提升损伤识别准确率的方式。

4.1.2 AlexNet for HMDI 四分类实验

(1) 基于少样本的四分类实验 (AlexNet for HMDI)

从原始未经数据增强的 5145 个训练样本中取出未损伤、裂缝、酥碱、剥落样本各 500 个，组成一个样本总数为 2000 的训练集。2000 个验证样本中每一类取出 50 个，组成一个样本总数为 200 的验证集。

将模型文件中最后的全连接层神经元数目改回 4，求解器文件中验证 batch 数目设置为 4，动量系数设置为 0.9，正则化系数设置为 0.0005，训练每迭代 20 次显示一次损失值，500 次进行一次验证，最大迭代次数为 100000。

训练耗时 9 小时 25 分 11 秒，实验结果如图 4.5 所示。

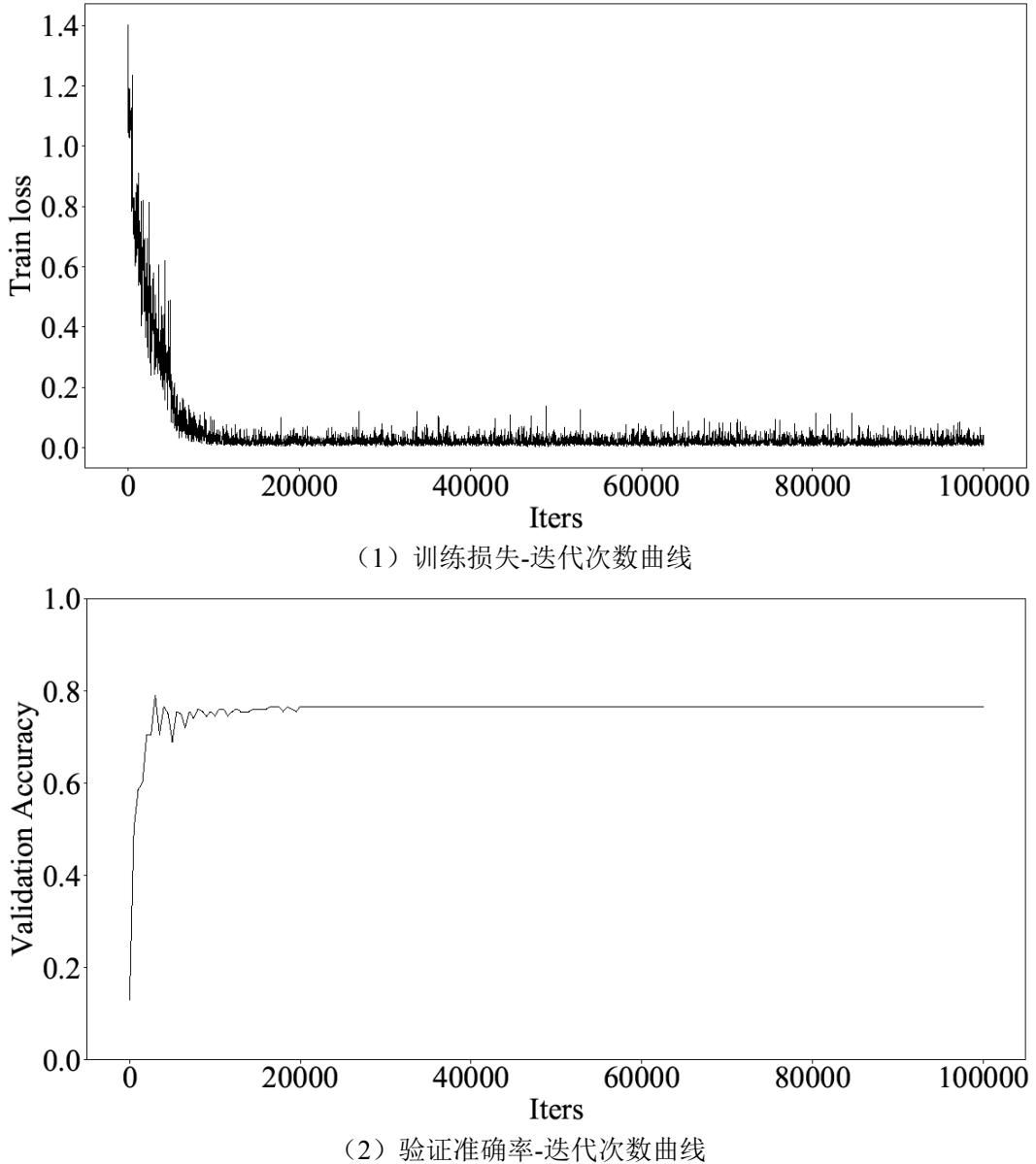


图 4.5 基于少样本的四分类实验结果 (AlexNet for HMDI)

由实验结果可知，训练损失值从 1.4 左右开始迅速下降，在 10000 次左右的迭代后趋于平缓。验证准确率从 25% 快速上升，在 4000 次左右的迭代后开始波动，到第 20000 次左右稳定于 76.5%。

对于损伤识别来说，这个准确率显然是不够的，而且相比于二分类的识别效果下降很多，说明在分类的过程中，不同损伤类型的识别较是否损伤的识别难度更大。在接下来的实验中，在保持网络结构不变的情况下，采取增大数据量的方式对识别准确率进行提升。

(2) 基于多样本的四分类实验 (AlexNet for HMDI)

这个实验使用全部的训练样本和验证样本, 即训练集包含未损伤、裂缝、酥碱、剥落样本各 10000 个; 验证集包含四类样本各 500 个, 基于上一个实验, 将求解器文件中的 batch 数目改为 40。

训练耗时 10 小时 2 分 16 秒, 实验结果如图 4.6 所示。

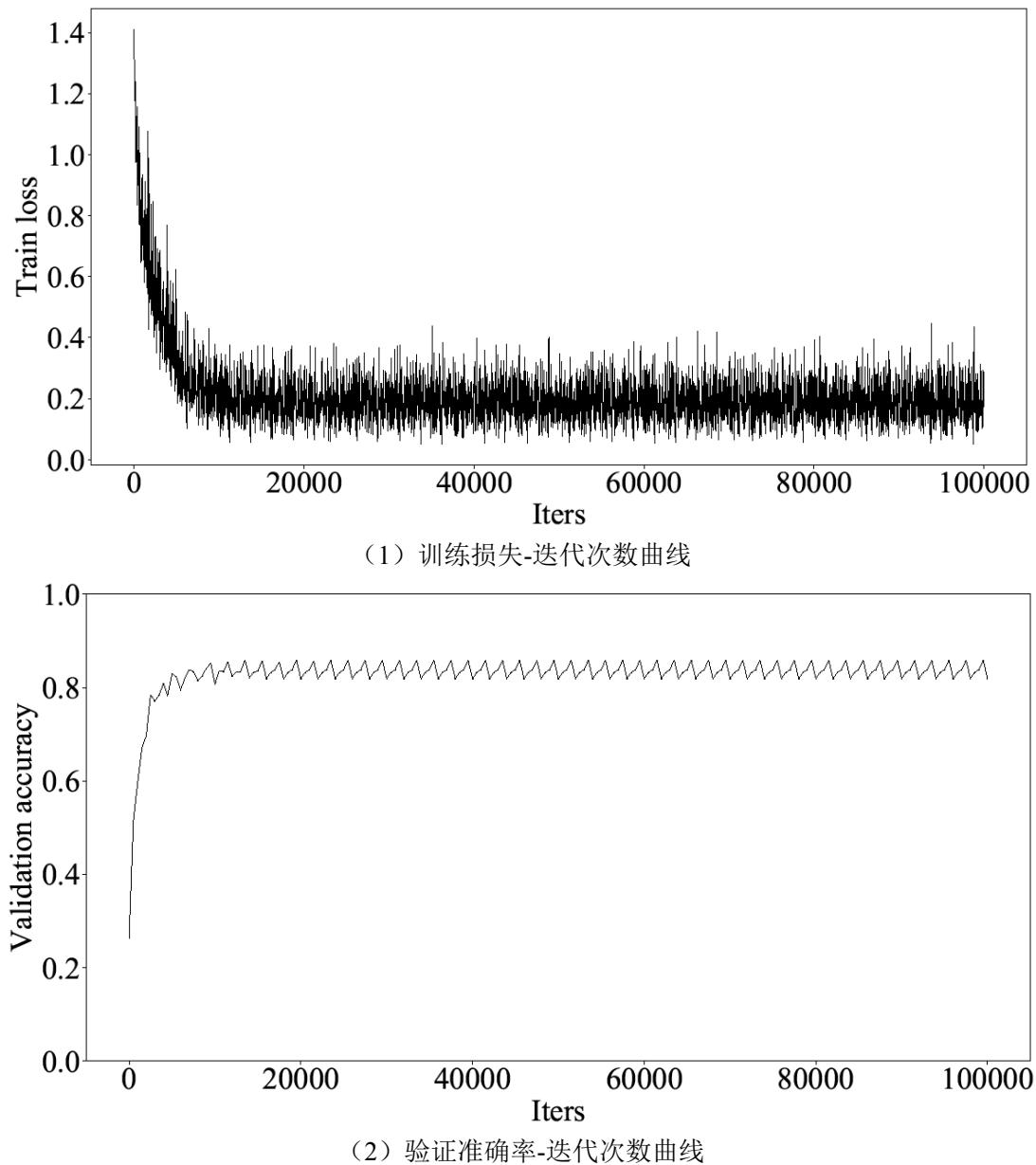


图 4.6 基于多样本的四分类实验结果 (AlexNet for HMDI)

由图 4.6 中的曲线可知，训练损失值从 1.4 左右迅速下降，在 5000 次左右的迭代后速度急剧下降，在 10000 次左右开始在 0.2 上下波动且幅度相对较大。对比图 4.4（1）可知，AlexNet for HMDI 在执行具有较多数据样本的训练时损失会出现相对较大的波动。验证准确率从初始的 25% 以较快速度上升，大约 4000 次迭代后出现了上下波动，20000 次左右处于一种“等幅振动”的状态：最高点为 83.4%，最低点为 80.9%。

此结果相比于少样本的四分类训练有一定的提高。然而对于实际的损伤识别任务，即使取本实验的最高准确率 83.4%，仍然不是一个令人满意的结果，并且最终准确率的较大波动也使分类结果无法具有较高的可信度。但是，通过这个实验确实可以说明，样本数量的增加，对于训练结果是具有一定正面影响的。进一步增加数据也许能够使准确率进一步的提升，然而，将数据扩充 20 倍仅仅提高了大约 6 个百分点，并且由本文 4.1.1 二分类的实验结果可知，准确率越高越难以提升。因此，本文对于准确率进一步的提升暂时不再采取更多的数据扩充，而是应用更深的网络模型，即 GoogLeNet for HMDI。

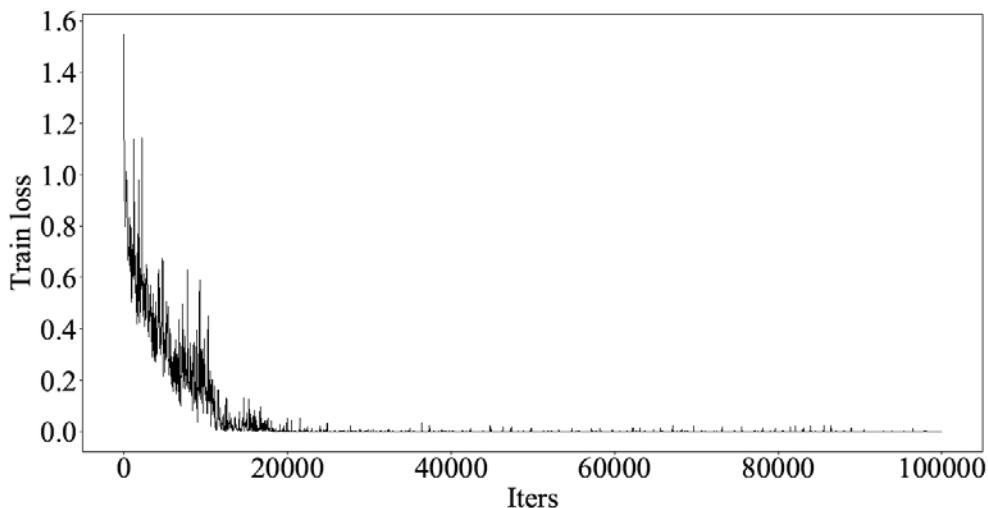
4.1.3 GoogLeNet for HMDI 四分类实验

（1）基于少样本的四分类实验（GoogLeNet for HMDI）

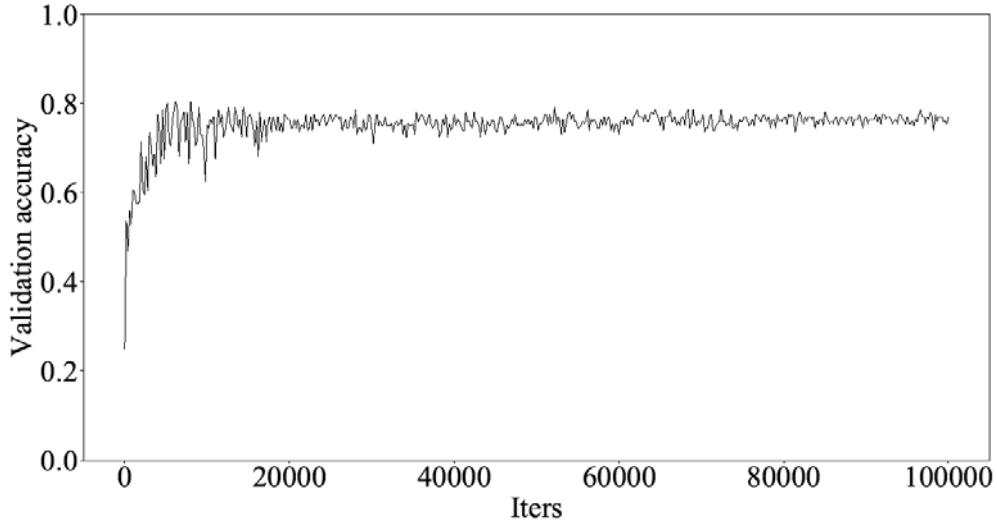
为了研究网络深度对于训练效果的影响并进一步提升分类器的准确率，在这一部分的实验中使用 GoogLeNet for HMDI 进行四分类的实验。本实验使用的训练集和验证集与 4.1.2（1）部分完全相同。

求解器文件中验证 batch 数目设置为 4，动量系数设置为 0.9，正则化系数设置为 0.0002，训练每迭代 40 次显示一次损失值，200 次进行一次验证，最大迭代次数为 100000。

训练耗时 16 小时 21 分 25 秒，实验结果如图 4.7 所示。



（1）训练损失-迭代次数曲线



(2) 验证准确率-迭代次数曲线

图 4.7 基于少样本的四分类实验结果 (GoogLeNet for HMDI)

由图 4.7 中的曲线可知，训练损失从 1.6 附近迅速下降，下降的过程中有较大的波动，在第 20000 次左右迭代时趋于平缓。验证准确率从 25% 开始上升，在第 20000 次左右迭代前波动较大，20000 次左右后波动较小，可以认为处于稳定状态。最终的平均验证准确率为 77.0%，显然这又是一个无法令人满意的结果，并且相对于使用同样数据样本的 AlexNet for HMDI 仅仅提高了 0.5 个百分点。因此在样本数量不够的情况下，网络的加深对识别准确率的提升效果并不是很明显。

(2) 基于多样本的四分类实验 (GoogLeNet for HMDI)

这个实验使用全部的训练样本和验证样本，即训练集和验证集与 4.1.2 (2) 部分完全相同。基于上一个实验，将求解器文件中的 batch 数目改为 40。

训练耗时 10 小时 2 分 16 秒，实验结果如图 4.6 所示。

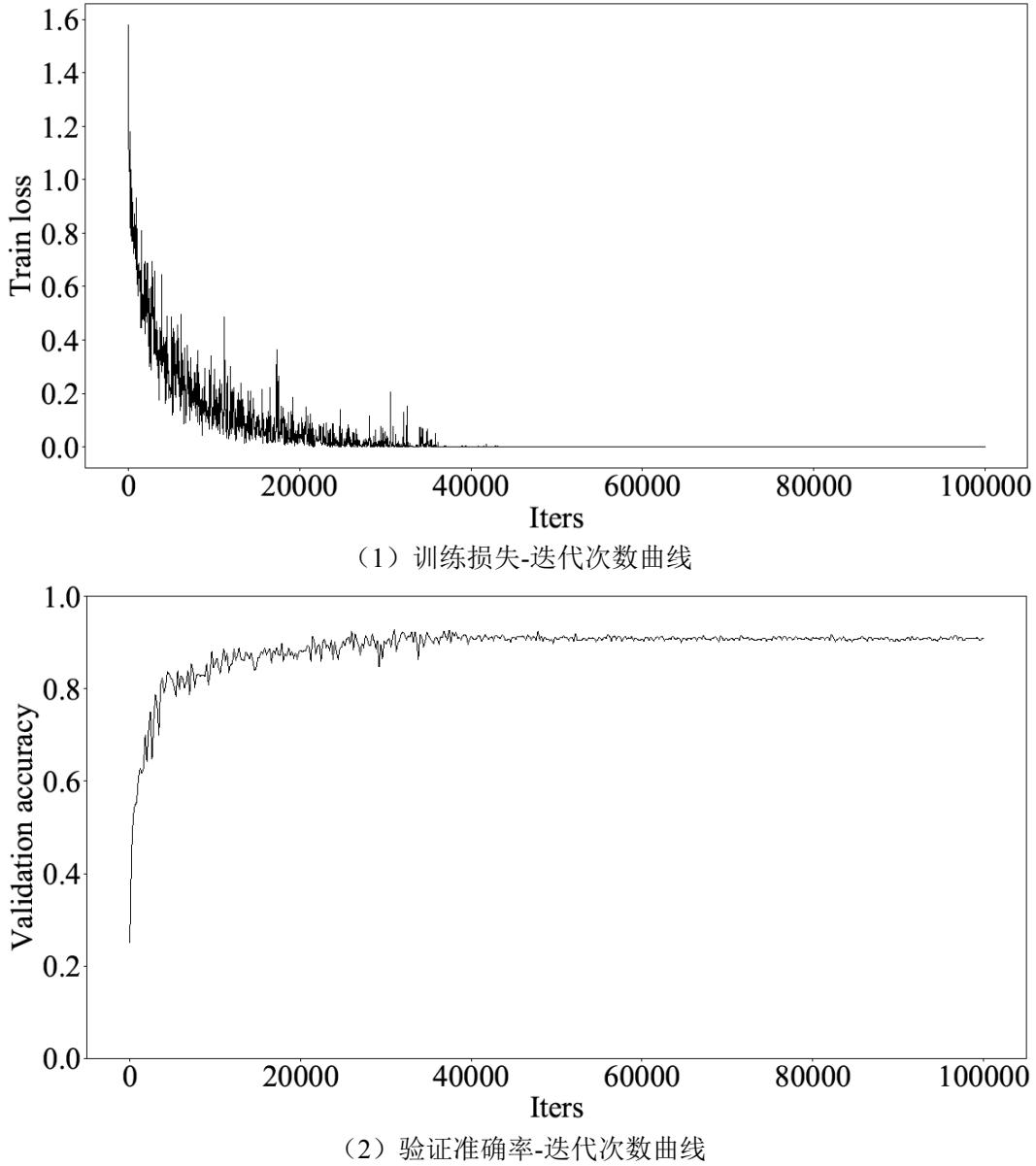


图 4.8 基于多样本的四分类实验结果 (GoogLeNet for HMDI)

由实验结果可知，网络的收敛与前几个实验相比较慢，因为无论是网络深度还是数据量的大小，这个实验都是最复杂的。训练损失从 1.6 附近开始下降，第 5000 次左右迭代后波动较大，到第 40000 次左右基本稳定。验证准确率从 25% 开始上升，第 5000 左右迭代后准确率提高速度立刻变慢且有些波动，到第 40000 次左右后波动微弱，可视为训练结果已经稳定。最终的平均验证准确率为 90.9%，对于四分类的损伤识别可以认为是一个合理的结果。

从以上所有的实验可知，任何情况下，数据量的增加对于识别效果的提升均具有很大的促进作用；在数据量充足的情况下，网络越深，识别效果越好；相同条件下的二分类比四分类训练效果好很多。

4.2 损伤定位技术

4.2.1 滑动窗口算法

4.1 部分为损伤识别技术的实现，然而在实际的工程中，仅仅识别是不够的，还要对不同的损伤进行定位，以便为后续的维修工作做指导。为了实现这一任务，本文在这一章中采用滑动窗口算法完成定位工作。

所谓的滑动窗口算法，既采取若干较小尺寸的“滑窗”对输入图像进行等步长或变步长扫描。滑窗每停留在一个位置，便将该位置的部分图像提取并输入训练好的分类器。每次测试选取一个损伤类别作为识别对象，若该位置的图像被分类器识别为这一类别（即结果为阳性），那么保留该位置的图像；若识别为其它类别（即结果为阴性），那么将该位置填充为白色。因此，最终的测试结果为白色背景上显示检测出的阳性区域，即完成了定位工作。

古建筑砌体结构表面的砖块一般由“顺”和“丁”组成，“顺”表示砖块的长边方向暴露在结构表面；“丁”表示砖块的短边方向暴露于结构表面。常见的砌筑方式有一顺一丁式、多顺一丁式、全顺式、顺丁相间式等等。本文实验中用到的故宫城墙样本为“顺丁相间”式：“顺”和“丁”相互交替，隔行对齐，如图 4.9 所示。

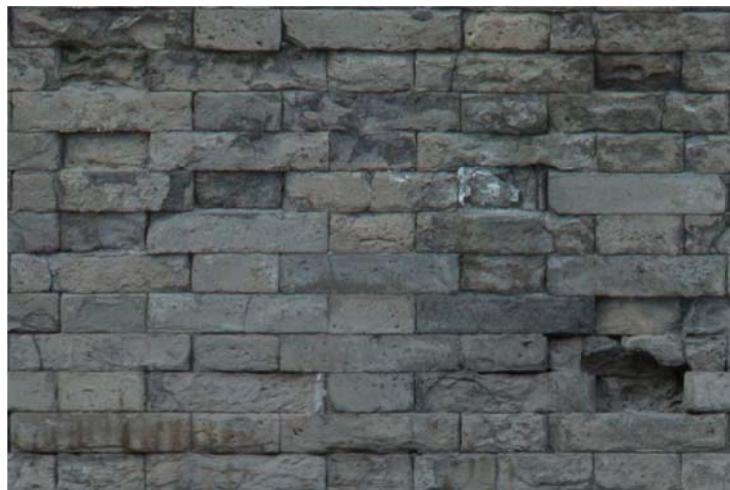


图 4.9 故宫城墙砌筑方式

由于砌体结构这种单元尺寸的特殊性，使用滑动窗口算法时采取两种不同尺寸的“滑窗”，即“顺”使用一种尺寸，“丁”使用一种尺寸。具体的数值根据实际的像素尺寸进行调节，并根据不同的砌筑方式采用不同的滑动策略，以达到全部扫描的目的。

对于本文的实验对象，即测试样本，采取每种窗口二次扫描的策略。由于砖块是顺丁交错且隔行对齐的，故每次扫描的过程中每次移动水平方向跨过一个砖块，竖直方向跨过一行。例如图 4.9 实际上为测试集中的一个样本，其像素分辨率为 1860×1260 ，“顺”滑窗尺寸为 480×105 ，“丁”滑窗尺寸为 210×105 。“顺”滑窗第一次扫描 xy 方向起始位置均设为 1；第二次扫描 y 方向起始位置为第二行，x 方向起始位置为该行第一个完整的“顺”的左上角，由数学关系可得，x 方向设为 $480-(480-210)/2+1=346$ ，y 方向设为 $105+1=106$ 。“丁”滑窗第一次扫描 y 方向起始位置为 1，x 方向起始位置需跨过一个“顺”，即 $480+1=481$ ；第二次扫描 y 方向起始位置为第二行，x 方向起始位置为该行第一个完整的“丁”的左上角，由数学关系可得，x 方向设为 $(480-210)/2+1=136$ ，y 方向设为 $105+1=106$ 。无论是“顺”滑窗或是“丁”滑窗，其每次 x 方向移动均为二者的 x 方向尺寸之和，即 $480+210=690$ ，y 方向移动均为二者该方向的尺寸，即 210，因此两种滑窗的移动步长均为 x 方向 690，y 方向 210，扫描策略如图 4.10 所示。

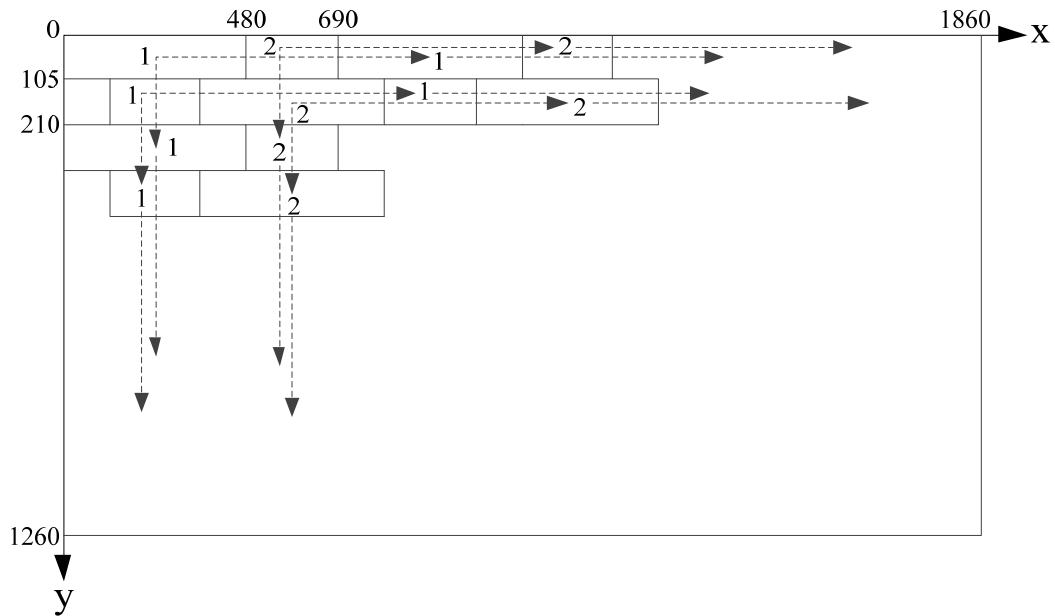


图 4.10 滑动窗口扫描策略实例

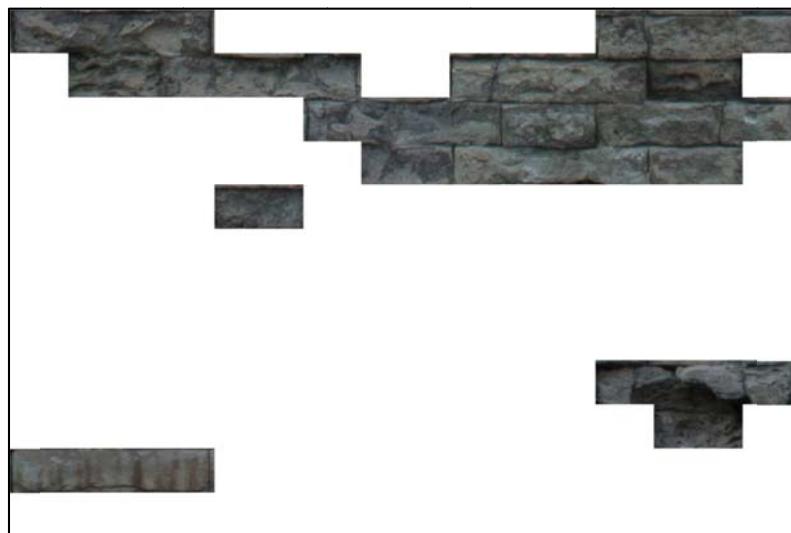
4.2.2 实验验证

按照上述的滑动窗口算法，利用 4.1 中识别准确率最高的训练好的四分类模型 GoogLeNet for HMDI (91%) 对测试集的 5 张图像进行扫描。为实现高效的可视化，测试结果采取分开显示的方式，即每一类损伤的识别结果单独显示于一张图像上。由于本文的研究为四分类，三种损伤，故最终每次的测试结果分开显示于 3 张独立的图像上。

对于每一类损伤的识别定位，将该类别定义为阳性，其他类别（包括未损伤）定义为阴性。根据测试结果，若某一砖块为阳性且被检测到，则称之为“真阳性”；若为阴性却被检测为阳性，则称之为“假阳性”；若为阳性且判断正确，则称之为“真阴性”；若为阳性却并未检测到，则称之为“假阴性”。

对于机器学习领域，常使用“准确率”、“精确率”和“召回率”三个指标来评价分类器的性能。首先，准确率很简单，即所有样本被正确分类的概率，反映了对整个测试集的样本识别能力，其表达式为：准确率=（真阳性+真阴性）/（阳性+阴性）。精确率为识别出的损伤中真正损伤所占的比重，反映了分类器能否精准识别的性能，其表达式为：精确率=真阳性/（真阳性+假阳性）。召回率为所有损伤中成功识别所占的比重，反映了分类器能否找出全部目标类别的性能，其表达式为：召回率=真阳性/（真阳性+假阴性）。

以一个测试样本（图 4.9）为例，该样本具有 20 个剥落砖块，7 个裂缝砖块和 1 个酥碱砖块，其余 32 个砖块未损伤，其实验结果如图 4.11 所示。



(1) 剥落测试结果



(2) 裂缝测试结果



(3) 酥碱测试结果

图 4.11 测试样本 1 实验结果

从实验结果可以看出，测试定位效果较好，能够直接显示损伤砖块的位置。关于识别结果，对于剥落损伤，4 块没有识别出来，其中 1 块误判成了裂缝，且有 1 块未损伤的砖块误判成了剥落；对于裂缝损伤，1 块没有识别出来，该砖块误判成了酥碱，且有 1 块剥落误判成了裂缝；对于只有 2 块的酥碱损伤，分类器成功将其识别，但是有 1 块裂缝误判成了酥碱。整体识别准确率为 90.0%，其中每一类的识别准确率为 91.7%、96.7% 和 98.3%。

该测试样本的错误分类绝大部分或者损伤程度较轻，或者具有少量另一类损伤特征，仅有一处错误分类为噪声干扰，其表面有一些深色的物质，被误判成了剥落，如图 4.12 所示。



图 4.12 噪声干扰造成的错误分类

尽管识别准确率仅达 90.0%，但是对于图 4.9 如此复杂的实际情况，分类器依然能够成功识别大部分的损伤区域。并且图像中很多砖块颜色差异较大，砖块边缘有时并不是很整齐，光照也并非严格均匀，这些均构成了噪声干扰。然而 60 个砖块中仅有 1 块是由噪声造成的分类错误，说明此方法的鲁棒性较高。

所有样本的测试结果见表 4.1。

表 4.1 样本测试结果

测试样本	整体准确率 (%)	损伤类别	阳性	阴性	真阳性	真阴性	假阳性	假阴性	准确率 (%)	精确率 (%)	召回率 (%)	备注
样本 1	90.0 (54/60)	剥落	20	40	16	39	1	4	91.7	94.1	80.0	图 4.11(1)
		裂缝	6	54	5	53	1	1	96.7	83.3	83.3	图 4.11(2)
		酥碱	1	59	1	58	1	0	98.3	50.0	100	图 4.11(3)
样本 2	95.0 (57/60)	剥落	6	54	5	53	1	1	96.7	83.3	83.3	—
		裂缝	3	57	2	57	0	1	98.3	100	66.7	—
		酥碱	5	55	5	54	1	0	98.3	83.3	100	—
样本 3	91.7 (55/60)	剥落	27	33	24	33	0	3	95.0	100	88.9	—
		裂缝	7	53	6	51	2	1	95.0	75.0	85.7	—
		酥碱	3	57	3	56	1	0	98.3	75.0	100	—
样本 4	91.7 (55/60)	剥落	18	52	16	50	2	2	94.3	88.9	88.9	—
		裂缝	4	56	2	55	1	2	95.0	66.7	50.0	—
		酥碱	4	56	3	56	0	1	98.3	100	75.0	—
样本 5	93.3 (56/60)	剥落	12	48	9	48	0	3	95.0	100	75.0	—
		裂缝	1	59	1	58	1	0	98.3	50.0	100	—
		酥碱	3	57	3	56	1	0	98.3	75.0	100	—

由测试结果可知，所有测试样本损伤识别的整体准确率均在 90% 以上，每一类的损伤识别准确率也均高于 90%，且大部分超过了 95%。同时，图 4.11 展示的是五个样本中相对结果最不好的一个，而这样一个样本仍然能够准确分类大部分损伤且几乎不受噪

声干扰，说明对于实际情况下的损伤识别定位工作，此方法具备很好的准确性和实用性。由于某些样本中的某类损伤较少，例如样本 1 中的酥碱损伤，样本 2 中的裂缝损伤等等，这些损伤一旦出现少量的分类错误，便会使精确率或召回率造成很大的下降，例如样本 1、4、5 中的某些指标最低达到了 50%。同基数增大的原理类似，当测试样本中该类别较多时这个问题将得以解决。

诚然，此方法仍具有一些局限性，例如样本中砖块并不是严格的整齐排列，一旦样本尺寸很大，这个问题便不能够忽略，此时有可能需一边进行滑动窗口，一边进行基线校准，并实施变步长微调。另外，若砖块尺寸有变，还应增加不同滑窗的数量；若结构的砌筑方式在某处发生变化，则又须对滑动策略进行调整。然而，对于古建筑来说，其表层绝大部分还是均匀整齐的，可以将整个结构图像进行分区处理，即将较大的图像拆分成相对较小的区域以便滑窗算法的实施。同时，对于特殊的部位（极其不规则的区域），可采用人工处理的方式，这样依然可以很大程度上提高效率，节约成本。

4.3 同数字图像处理方法的对比研究

数字图像处理仍是最常见的视觉方法，其主要依靠边缘检测算法提取特征，而不是通过大量样本的训练自动提取。本文的 1.2.2 部分已经大致说明了主要的数字图像处理方法及其基本原理，在这一部分中，使用几个最常用且经典的边缘检测算法对图 4.11 的测试样本进行处理，并对其结果进行分析。

本文使用 Sobel、Robert、Prewitt 和 Canny 四种算子分别对样本进行处理，结果如图 4.13 所示。



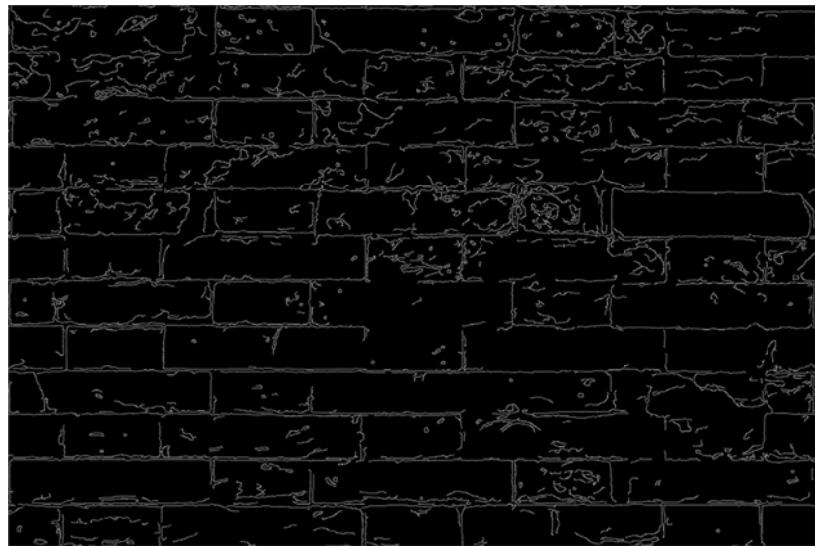
(1) Sobel 边缘检测结果（阈值取 0.4）



(2) Robert 边缘检测结果（阈值取 0.09）



(3) Prewitt 边缘检测结果（阈值取 0.3）



(4) Canny 边缘检测结果 (阈值取 0.3)
图 4.13 采用四种边缘检测算法处理测试样本

四种边缘检测算法均使用 MATLAB 自带的函数实现，阈值参数的取值通过实验的最优效果得到。由图可以看出，Sobel、Robert 和 Prewitt 三种一阶算法效果类似，且对噪声相对敏感。这三种算法均能区分砖块的完整轮廓，以及损伤较为严重的部位，如大块的剥落。然而对于细裂缝等小范围损伤，这三种算法均无法给出有效的识别信息，既无法从噪声中有效提取特征。由于材质和光照的影响，这三种算法处理后的图像的上半部分干扰因素很多，其中 Sobel 算法相对好一些，但鲁棒性均不高。Canny 算法相比于上述其它三种算法效果较好，其采用高斯滤波有效降低了噪声的干扰，且在识别裂缝损伤时显示了较好的性能。然而 Canny 算法仍然无法区分材质上的噪声干扰因素，且无法判定“酥碱”这一类别，即对于损伤的分类，仍达不到合理的检测要求。

对比图 4.13 和图 4.11，基于深度学习方法的损伤检测相比于单纯的图像处理显示了巨大的优越性。在卷积神经网络的训练中，分类器会自动地学习图像特征，极大地提高了效率。并且，若没有精细参数化的方法，使用图像处理技术找到的也许并非某一类别的损伤，甚至仅仅是噪声干扰。相比之下，深度学习方法是从大量图像中学习其不变的特征，若某样本分类错误，仅需将该样本加入到训练集继续进行网络训练，从而使分类器的性能越来越强，这些属性使得此方法具有很高的准确性和鲁棒性。

4.4 本章小结

本章以北京故宫城墙为例，详细说明了古建筑砌体结构损伤识别与定位的实现，并通过大量实验验证，结果表明该方法具有准确、高效和实用的特点。相比于图像处理，

深度学习方法无需人工的特征提取，且受噪声影响很小。对于损伤识别性能的提升，主要采取数据量的扩充和网络加深两种方法，二者提升效果均很明显，在数据量充足的情况下改善网络是一个很好的选择。同时，超参数的选择对于识别效果来说也很重要。对于滑动窗口算法，需注意滑窗尺寸的选取以及扫描策略，这也是此方法的一个缺点。当图像尺寸过大或出现不规则的区域时，使用滑动窗口算法效率往往不是很高，还需采用更多滑窗、变步长以及分割图像等方式进一步地处理。因此，下一章将采用全新的基于候选区域目标的深度学习损伤识别方式来解决这些问题。

5 基于候选区域目标的深度学习损伤检测技术

5.1 方法原理

5.1.1 RCNN 和 Fast-RCNN

由前文可知，单独的卷积神经网络只能解决分类问题。对于目标的定位和提取，仍需要借助一些例如滑动窗口算法的额外手段完成工作。例如本文的损伤识别，若使用滑动窗口算法需要对每一块砖块进行扫描，对于不同的组砌方式，不同的单位像素尺寸，以及不规则的区域位置，还需实施不同的扫描策略，设计不同的算法，对于特别大尺寸的结构表层图像来说还需以人工的方式对样本进行预处理，这些问题都不可避免地限制了深度学习方法的效率和实用性。因此，本章进行初步的研究和实验，应用基于候选区域目标的深度学习算法来尝试解决上述问题。

所谓的候选区域目标，就是在对损伤进行识别时，根据目标类别在整体图像中的局部特征体现，直接提前找出该类别目标可能出现的区域，即“候选区域”。对这些区域实施进一步的处理并输入卷积神经网络，成功识别后对目标范围框实施回归算法，更加精确地完成定位任务。

目标检测的第一个算法是 2014 年由 Girshick 等学者提出的 RCNN 算法^[113]，R 表示“Region”，即区域。该算法首次将目标区域的定位任务集成于深度学习当中，是深度学习检测领域的开山经典之作，目前几乎所有的目标检测算法都继承和发展了 RCNN 的思想。RCNN 算法分为 3 个步骤：第一步使用选择性搜索（Selective Search）算法^[114]获得候选区域；第二步与卷积神经网络的分类任务相同，即将候选区域输入训练好的卷积神经网络模型计算类别评分的向量并利用 SVM 进行分类，第三步使用回归算法对目标边框进行进一步的修正（平移、缩放），从而精确地完成目标类别的定位任务。

这里说明一下第一步。选择性搜索算法在前期使用 Felzenszwalb 和 Huttenlocher 提出的图像分割算法^[115]将原始输入图像进行区域分割，即以像素作为区域的“顶点”，根据相邻像素之间关系确定区域的“边”，应用一定的边缘判别方式（阈值函数）从而初步将整个图像分割成大量较小的区域。初步获得分割区域后基于颜色、纹理、尺寸等特征计算相邻区域的之间相似度，将两两最大相似度的区域合并成一个新的区域并再次计算相邻相似度，如此循环，直至将整张图像合并成一个区域，在这个过程中，所有曾经出现过的区域即为候选区域。

需要注意的一点是，在分类后为了消除多余的目标选框，RCNN 算法采取非极大值抑制的方法，其使用了检测评价函数（IOU），它表示了两个目标选框的交集占并集的比例，如图 5.1 所示。

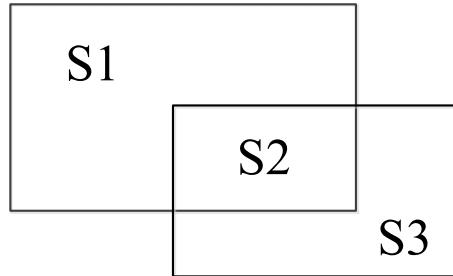


图 5.1 两个具有重叠部分的目标选框

则 IOU 表示如下：

$$\text{IOU} = \frac{S_2}{S_1 + S_2 + S_3} \quad (5.1)$$

例如对故宫城墙样本的某一部分图像进行表层损伤检测任务，假设经过卷积神经网络评分后具有 5 个相互重叠的目标选框 A、B、C、D、E，如图 5.2 所示。

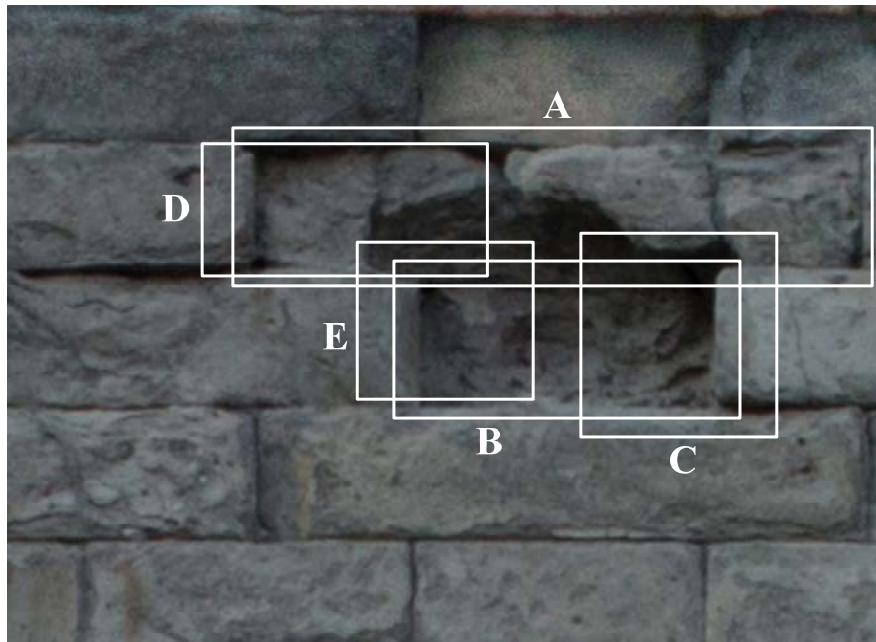


图 5.2 多余目标选框消除示例

若 5 个目标选框的损伤类别评分由大至小的顺序排列为 A>B>C>D>E。从类别评分最大的 A 选框开始，评分小于 A 的选框中，D 与 A 的 IOU 值超过了设定的阈值，因此将 D 删除。继续使用这个方法分析类别评分第二大的 B 选框，评分小于 B 的选框中，C 与 B、E 与 B 的 IOU 值超过了设定的阈值，因此将 C 和 E 删除。继续分析，此时已无其它目标选框，因此，A 和 B 即为目标结果。

然而 RCNN，包括之后针对 RCNN 只能输入特定尺寸图像问题而改进的 SPP-Net^[116]，都具有三个缺陷。第一，训练效率较低：RCNN 的三个步骤（获取候选区域、分类、回归）属于三个阶段，即训练时必须首先运用选择性搜索算法得到候选目标，然后输入卷积神经网络进行训练，最后完成回归任务。这三个阶段必须先后进行，耗费大量的训练时间。第二，训练数据占用空间过大：由于分类任务和回归任务先后独立进行，因此需要两个卷积神经网络进行训练，图像特征占用的存储空间增加了一倍。第三，测试（检测）效率较低，原因与第一个缺陷相同，同样需要三个步骤先后分开执行。

针对上述问题，Girshick 提出了 Fast-RCNN 算法^[117]，即较快版本的 RCNN。Fast-RCNN 在很大程度上克服了 RCNN 和 SPP-Net 的三个缺陷，首先，它在最后一个卷积层的后面加入了一个叫做感兴趣区域（ROI）汇聚层。ROI 实际上就是 RCNN 第一步使用选择性搜索算法得到的候选目标坐标在卷积后特征图上的映射，即与 RCNN 将候选区域输入卷积神经网络不同，Fast-RCNN 将整张图像以及候选区域的坐标作为输入，经过最后的卷积之后将候选区域映射在特征图上再进行提取，从而节约了大量的训练和测试时间。同时，Fast-RCNN 将 RCNN 中的 SVM 用 Softmax 进行替代，并将分类任务和回归任务融合在一起，使用承担两个任务的损失函数从而共享一套卷积神经网络进行训练，大大节约了图像特征存储空间，其算法架构如图 5.3 所示。

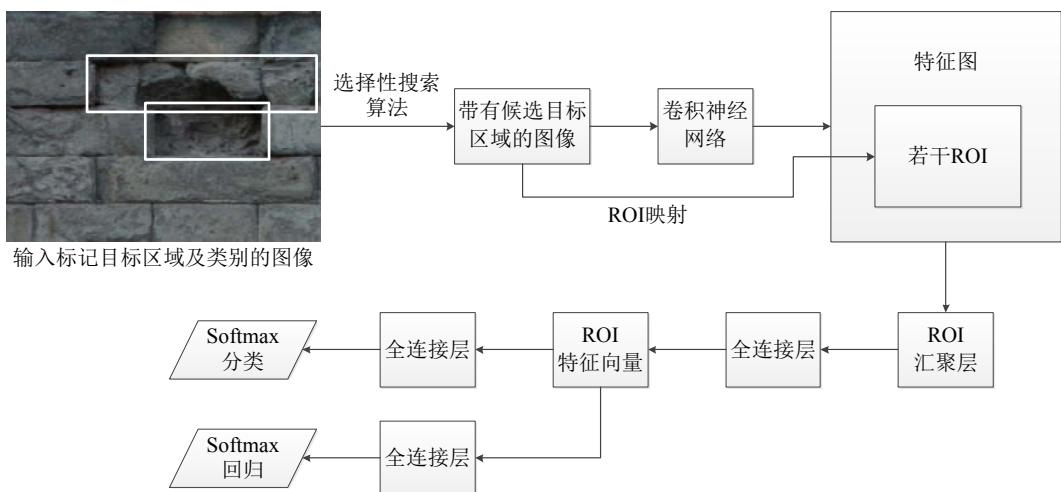


图 5.3 Fast-RCNN 算法架构

Fast-RCNN 很大程度上解决了 RCNN 和 SPP-Net 中存在的问题。然而，Fast-RCNN 中的候选目标的提取，即选择性搜索算法与卷积神经网络的训练和测试仍然是分开进行的，而实施选择性算法所使用的时间占用了训练过程的绝大部分（通常候选特征的提取时间是分类回归所用时间的好几倍^[117]），将这一步骤加入到卷积神经网络中实现直接点对点的训练和测试是解决此问题的关键，下一部分的 Faster-RCNN^[118]，即更快版本的 RCNN，成功实现了这一目的。

5.1.2 Faster-RCNN

Faster-RCNN 是 2015 年由任少卿在微软亚洲研究院实习期间与何凯明、Girshick 和孙剑共同开发的目标检测算法^[118]，它成功在 Fast-RCNN 的基础上，将候选区域的获取集成在了卷积神经网络中，实现了点对点的训练与测试功能，效率成倍提高，几乎达到了实时检测的水准。直到现在，综合考虑其速度和准确性，Faster-RCNN 仍是目标检测领域应用最广泛，最被认可的算法。

Faster-RCNN 的技术核心是一个称为“候选区域网络（RPN）”的结构，该结构属于卷积神经网络的一部分，承担着提取候选区域的任务，取代了 RCNN 和 Fast-RCNN 中的选择性搜索算法。不同于 RCNN 和 Fast-RCNN，Faster-RCNN 并不是先提取候选区域再输入卷积神经网络，而是直接将整张图像输入到卷积神经网络中，经过一系列的卷积和汇聚操作得到该图像的特征图。之后才加入 RPN，即通过对特征图的操作处理来获取 ROI。

RPN 中提出了一个称为“锚”的概念。所谓的锚，是若干在原图上具有特定尺寸和宽高比的矩形区域框，在 RPN 中，锚的种类数量 k 以及各个种类的尺寸和宽高比是可以改变的超参数。在 Fast-RCNN 论文中，作者将锚只定义了三个宽高比：1:1、2:1 和 1:2，其中每个比例具有三个不同的像素尺寸，这样锚的种类 k 即为 9。对于特征图上的每一个点，必然会在原图上具有映射，假定以该点的映射为中心，构造上述 9 种类型的矩形框，那么这 9 个矩形框即为该点的锚。例如图 5.4，若特征图的尺寸为 6×6 ，那么输入图像中锚的个数即为 $6 \times 6 \times 9 = 324$ 个。

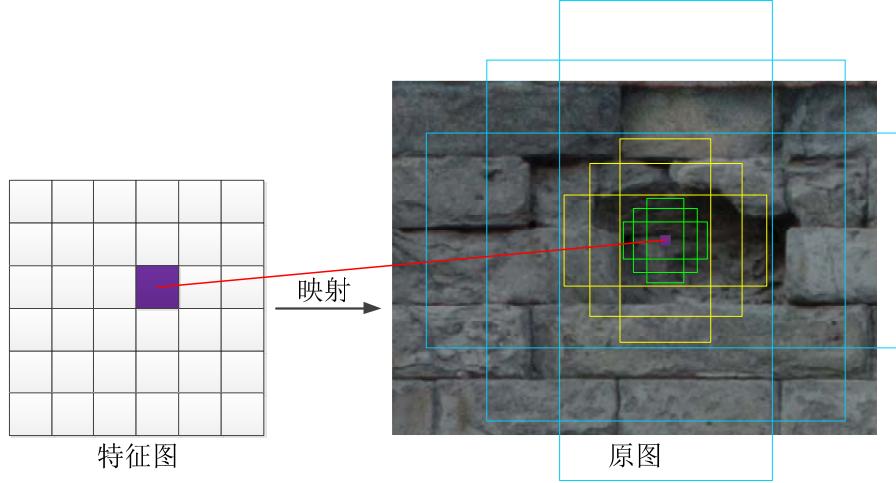


图 5.4 特征图中某一点对应原图的 9 个锚

根据不同的检测任务以及目标的尺寸，需合理地选择锚的尺寸和比例，一般来说，1:1、2:1 和 1:2 三个比例几乎能够满足任何目标检测的要求，根据目标在原图中的可能大小，选择合适的锚的尺寸即可，即使某些目标的比例和尺寸不在锚的范围内，RPN 中的边框回归也能够轻松地解决问题。

需要说明的是，并不是所有的锚均参与 RPN 的训练。RPN 的目标是获取候选区域，而在训练中很多锚与实际的目标并没有交集，因此为了提高训练效率，需对锚进行进一步的筛选。RPN 中将所有目标类别合并为一类；背景算作另外一类。对于所有的没有超出图像边界的锚，计算其每一个与训练样本中标记的目标选框的 IOU，若 $\text{IOU} > 0.7$ ，则判定为“目标”；若 $\text{IOU} < 0.3$ ，则判定为“背景”，其余的锚全部舍弃。

经过筛选的锚便可以进行训练，将锚在特征图上的对应点进行卷积运算。具体方式为使用一个 3×3 的卷积核，其中心为锚的对应点，这样，卷积核每停留在一个位置便会输出一个神经元个数为特征图深度方向尺寸的特征向量。例如当卷积神经网络使用 ZF-Net 时，特征向量的尺寸即为 256×1 。同一点的每个参加训练的锚的特征向量相同，对于分类损失，该向量与一个 2×1 的向量全连接，其分量表示“目标”和“背景”；对于回归损失，该向量同时与一个 4×1 的向量全连接，其分量表示“ x 方向平移”、“ y 方向平移”、“ x 方向缩放”和“ y 方向缩放”。

因此，RPN 的损失函数如下：

$$L = \frac{1}{N_{\text{cls}}} \sum_i L_{\text{cls}}(p_i, p_i^*) + \lambda \frac{1}{N_{\text{reg}}} \sum_i p_i^* L_{\text{reg}}(t_i, t_i^*) \quad (5.2)$$

该损失函数包含两部分：cls 表示分类，reg 表示回归。 N_{cls} 为参加训练的锚的总数， N_{reg} 为锚的位置信息总数。 λ 为平衡系数，是一个超参数，目的是令分类损失和回归损失所

占的比重大致相等。 p_i 为第 i 个锚为“目标”的概率（Softmax 计算）， p_i^* 为该锚经 IOU 判定后的实际类别：“目标”的 p_i^* 值为 1；“背景”的 p_i^* 值为 0。 t_i 和 t_i^* 各分为 4 部分（两个方向的平移和缩放），计算如下：

$$t_x = (x - x_a) / w_a; \quad t_y = (y - y_a) / h_a; \quad t_w = \log \frac{w}{w_a}; \quad t_h = \log \frac{h}{h_a} \quad (5.3)$$

$$t_x^* = (x^* - x_a) / w_a; \quad t_y^* = (y^* - y_a) / h_a; \quad t_w^* = \log \frac{w^*}{w_a}; \quad t_h^* = \log \frac{h^*}{h_a} \quad (5.4)$$

x 、 y 、 w 、 h 分别表示中心的 x 坐标， y 坐标， x 方向的长度尺寸和 y 方向的长度尺寸。含有“*”上标的为最近目标的实际位置；含有“a”下标的为锚的位置；不含上下标的为预测位置（即训练位置）。

L_{cls} 和 L_{reg} 计算如下：

$$L_{cls}(p_i, p_i^*) = -\log[p_i^* p_i + (1-p_i^*)(1-p_i)] \quad (5.6)$$

$$L_{reg}(t_i, t_i^*) = \begin{cases} 0.5(t_i - t_i^*)^2 & (|x| < 1) \\ |x| - 0.5 & (|x| \geq 1) \end{cases} \quad (5.7)$$

RPN 训练完成后便获得了 ROI，接下来的步骤和 Fast-RCNN 完全相同，即 Faster-RCNN 相当于 RPN 与 Fast-RCNN 的组合。其算法架构如图 5.5 所示。

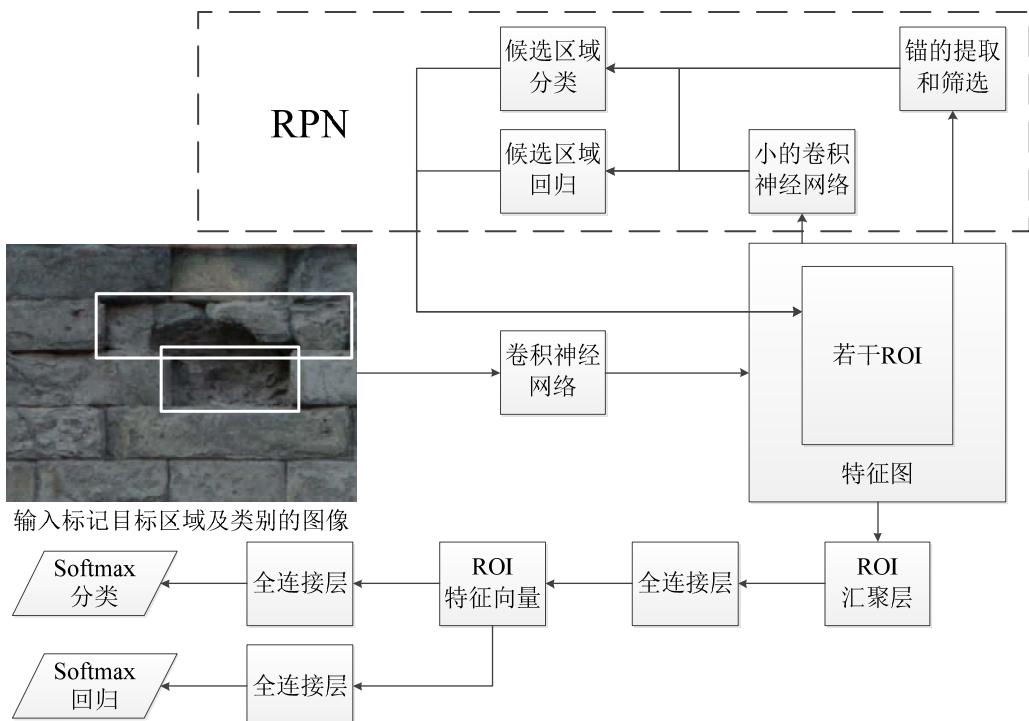


图 5.5 Faster-RCNN 算法架构

5.2 基于 Faster-RCNN 的损伤检测实验

这一部分基于 Faster-RCNN 算法，应用第三章建立的数据集，使用其配置好的训练工具和环境，对北京故宫城墙采用全新的深度学习方法进行损伤检测实验。

训练及测试环境仍然时用搭建好的 Caffe 框架，只是由于将识别和定位任务结合在一起，需要在测试时直接对结果进行可视化处理，因此为了方便，实验使用 Caffe 的 MATLAB 接口。Faster-RCNN MATLAB 版本的源代码在 Github 上取得^[119]，并在工作站上进行调整和编译。

首先，将数据集制作成 PASCAL VOC 格式^[120,121]便于网络的训练，将 260 个样本统一命名为连续的数字，并使用 Github 上的 labelImg 工具^[122]对样本进行损伤标注，生成.xml 格式文件。最后随机从这 260 个样本中提取 30 个作为验证样本，30 个作为测试样本，其余 200 个作为训练样本，将训练集、验证集、测试集各自样本的文件名保存于各自的文本文件中。至此，数据集的准备全部完成。

将 Fast-RCNN 按照工作站的软硬件条件进行调节和编译，并配置所有的文件路径和超参数。由于训练样本较少，故 Fast-RCNN 中的卷积神经网络模型选用 ZF-Net^[71]，其相比于 AlexNet 层数没有变化，只是基于 AlexNet 进行了一些微调。由于本实验并不区分损伤的类别，因此可以认为只是一个“目标”与“背景”的二分类问题，从而需修改源代码中有关结构层的输出神经元个数。求解器文件中基础学习率设置为 0.001，学习率衰减系数设置为 0.1，衰减步长设置为 30000，动量系数设置为 0.9，正则化系数设置为 0.0005，最大迭代次数为 40000。

所有准备工作就绪后，开始训练。训练耗时 7 小时 34 分钟，成功获得用于损伤检测的 Faster-RCNN 模型，测试集的精确率与召回率关系曲线如图 5.6 所示。

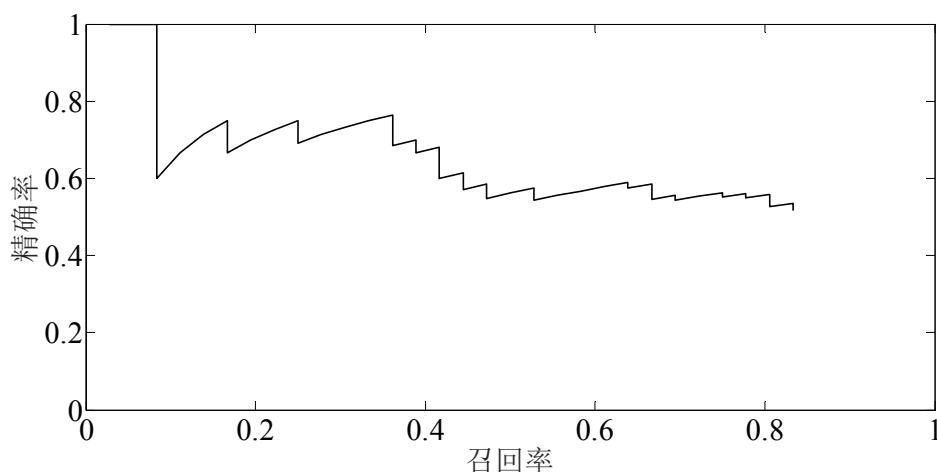


图 5.6 测试集精确率-召回率曲线

通过自动调节网络中的阈值，获得一组不同的精确率和召回率的值。对于包括深度学习在内的机器学习训练，二者的值越接近于 1 越好。然而实际上情况下，精确率和召回率往往此消彼长，即精确率提高，则召回率降低，反之亦然。在召回率较高的情况下（80% 左右），精确率仅在 58% 左右。对于本文特定的损伤检测任务，由于要尽可能地“找全”，即使有时检测到的可能并不是损伤，也好于真正的损伤没有被检测到。因此，对于本文来说，“召回率”更加重要一些。

导致精确率不是很高的最主要原因就是训练样本太少，一般来说，PASCAL VOC 和 ImageNet 等数据集的样本数都是以万为单位的，尽管这些数据集的分类数目也很多，但是扩充样本是提高本实验识别效果的最大可能的方式，在未来的研究中，将会继续完成这一工作。

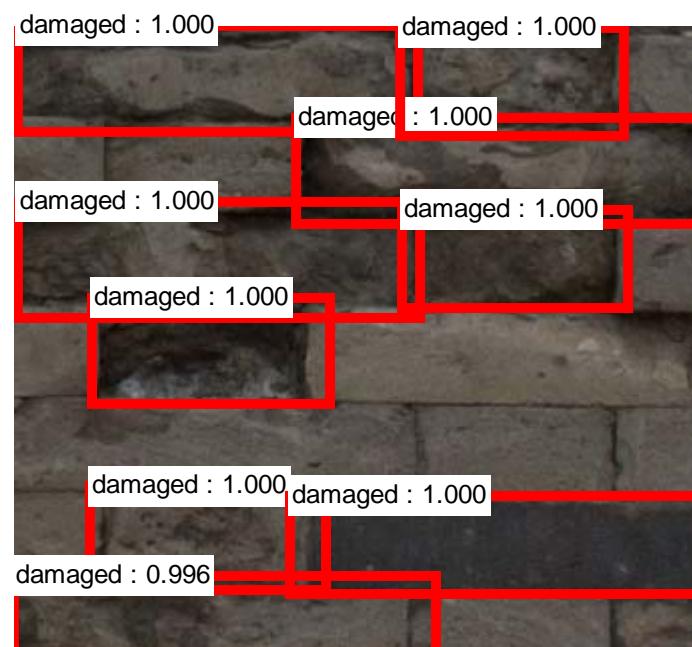
提取训练好的 Faster-RCNN 模型，判定阈值设为 0.7，并从原始城墙图像样本中另截取 6 个不同于实验使用过的数据集区域的图像样本。其中 3 个样本和实验数据样本所用的尺寸相同，为 760×700 ；另外 3 个样本尺寸比实验数据样本大，为 1250×1250 。每张图像的检测时间均在 0.05s 左右，检测结果如图 5.7 所示。



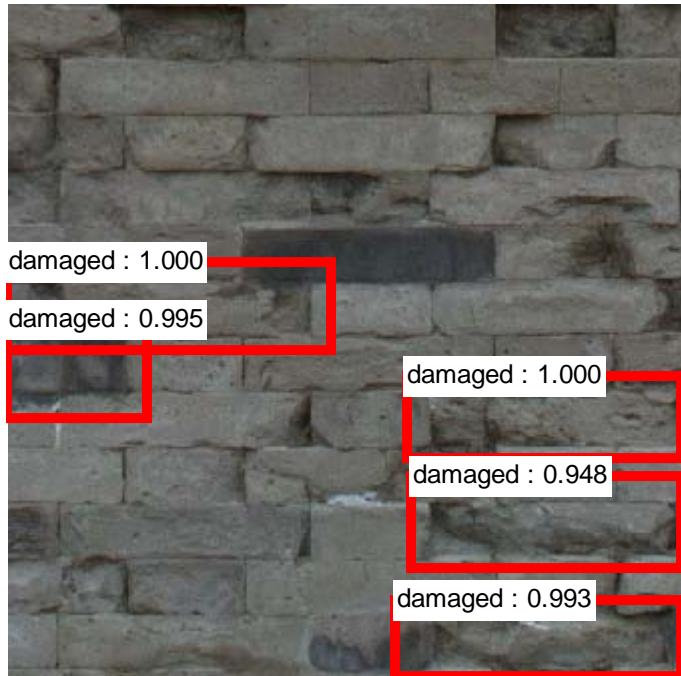
(1) 小尺寸样本 1 检测结果



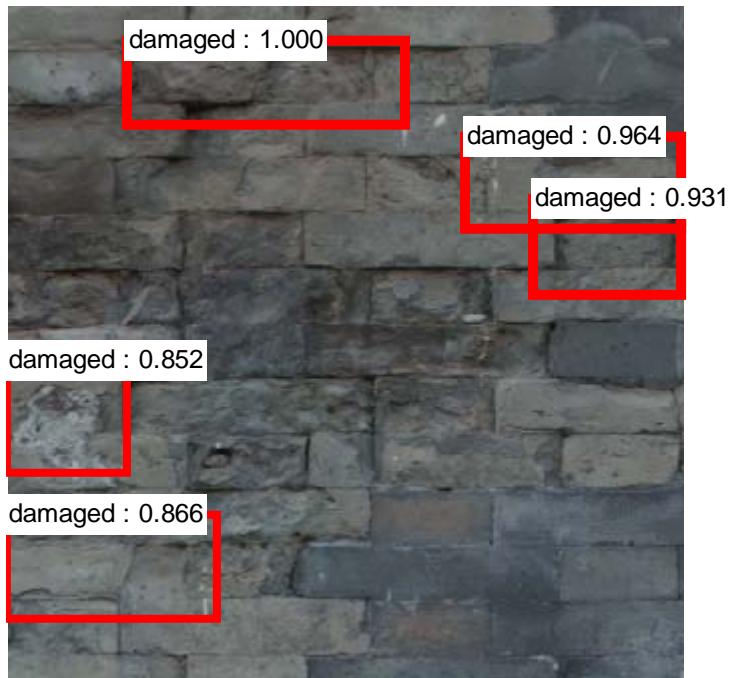
(2) 小尺寸样本 2 检测结果



(3) 小尺寸样本 3 检测结果



(4) 大尺寸样本 1 检测结果



(5) 大尺寸样本 2 检测结果



(6) 大尺寸样本 3 检测结果

图 5.7 样本检测结果

由结果可知，小尺寸的样本损伤检测效果较好，几乎能够将样本中所有的损伤砖块全部识别定位成功，且错误识别较少；大尺寸样本损伤检测效果较差，原因可能为训练集中的样本尺寸过于单一，导致检测并没有较好的适应性。在未来的研究中，应采取较多尺寸的样本进行训练，并对数据集进行大量扩充，从而解决精确率和适应性的问题。

5.3 本章小结

本章提出了一种以 Faster-RCNN 算法为核心的全新的基于深度学习目标检测方法的古建筑砌体结构损伤识别与定位技术，该技术将结构损伤的识别与定位技术结合在一起，无需通过滑动窗口，能够对任意尺寸的测试样本进行损伤检测。然而，由于样本数量太少且尺寸较为单一，实验并没有达到特别好的效果。实际的检测对于同训练集图像尺寸较为接近的样本效果很好，对于更大的图像检测效果较差，在未来的研究中应通过对样本及其尺寸类型的扩充解决这一问题。

然而，实验的结果足以说明这种方法的潜在可行性和巨大的发展空间，通过扩充样本成功提高识别精确率后，此方法将发挥巨大的高效性和实用性。为了解决样本不足的问题，本文在接下来的最后一章中初步提出了“众包式损伤检测”思想，为结构损伤大数据的采集提供了一个新的思路。

6 众包式古建筑损伤检测模式初探

6.1 众包式古建筑损伤检测模式

人工智能领域之所以能够持续升温并且发挥巨大的作用，除了因为计算机硬件性能不断地提升和 GPU 加速训练的实现，很重要的一点就是海量数据的支持。由第四章和第五章的训练和测试实验可知，数据样本的数量对于损伤检测效果来说是极其重要的，如果能够获取更多的古建筑结构表层图像训练样本，甚至建立一个庞大的损伤检测数据库，即“结构损伤大数据”，无论对于损伤检测分类器的性能提升，还是人工智能技术在土木工程领域应用的发展，都是十分有利的。

因此，为了在未来的工作中能够实现“结构损伤大数据”的高效采集，本文初步提出了一种“众包式古建筑损伤检测模式”。“众包”一词由 Jeff Howe 于 2006 年首次提出^[123]，其含义为“专业的公司或机构将特定的任务通过自愿的方式外包给大众去完成（通常是以互联网的形式）^[124]”。在土木工程结构健康领域，这种众包的思想在很多研究和应用中均有体现^[125,126]，而本文的“众包”，即为让公众参与到建立“结构损伤大数据”的工作当中。由第三章可知，本文的研究和应用所使用的数据样本均为图像的形式，因此，只要具备任何拍摄图像的条件（如智能手机，数码相机等等），任何人都能够实现数据样本的采集，若大众以互联网为渠道将获取的数据样本进行上传，便能够对数据库的扩充作出巨大的贡献。

需要说明的是，公众在这个过程中只是将拍照得到的图像进行上传，并不会参与到数据的处理甚至分类的工作。因此，该过程无需任何的专业性，只要具备拍照条件和互联网环境，任何人均可以实现这一过程。通过建立服务器和网站，必要时施行一些奖励机制，号召大众在古建名胜游玩时，拍摄其近景图像进行上传，后续的工作由工作或专业人员进行完成。尽管这一收集数据的过程看似简单容易，但是大量数据的获取若仅靠少数人员去完成，不仅效率很低，其工作量也是巨大的。众包模式的好处在于，专业人员无需花费大量的时间和精力经常赶往现场拍摄图像，这些时间和精力可以用于后续的数据处理、分类标记以及进一步的研究工作。

在未来的研究中，将会搭建公众用于上传图像样本的网站以及用于储存样本的服务器，这个过程也可以成为“云数据”的建立。原始样本获取后，由工作人员对上传的图像进行对训练、验证和测试样本提取和初步处理，之后再由专业检测人员对处理后的样本进行分类和标注工作，使样本成为标准的数据集。最后不断地将新的数据集加入“结构损伤大数据”，并对网络模型进行训练和改进，优化其性能。该模式流程如图 6.1 所示。

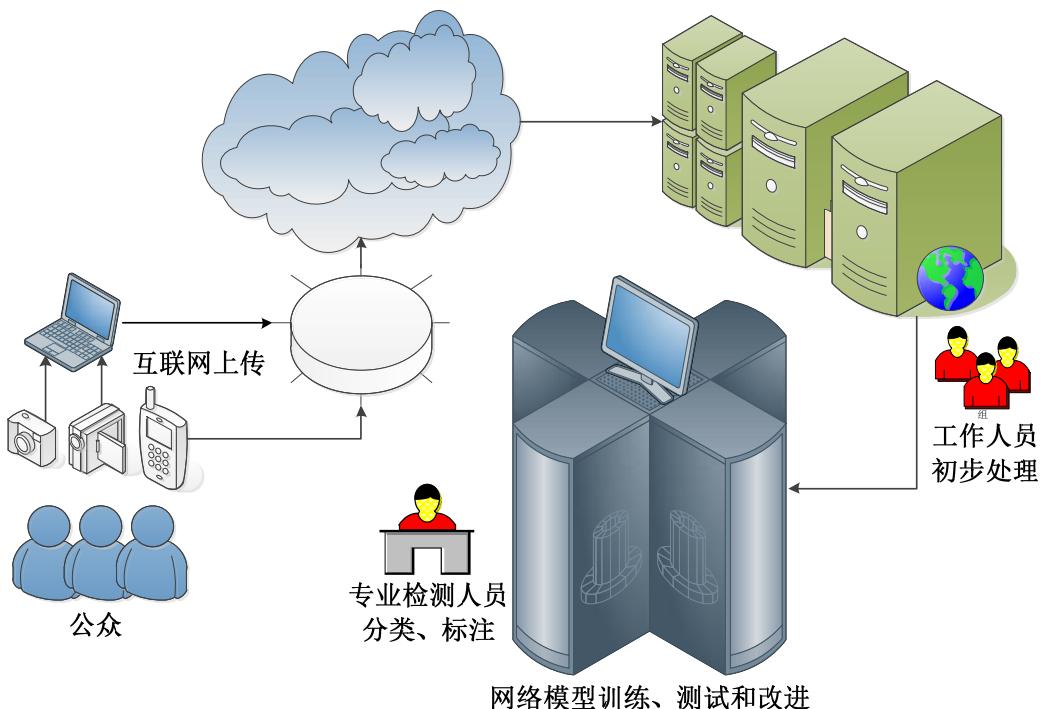


图 6.1 众包式损伤检测模式

6.2 本章小结

本章初步提出了一个众包式古建筑损伤检测模式，通过调动公众收集图像样本，后续由工作人员处理，专业人员分类和标注，从而快速高效地增加用于训练和测试的样本数量，有望实现“结构损伤大数据”的建立。

采取众包模式可以大量减少数据收集的时间，提升效率。并且大数据的实现使得更深卷积神经网络模型的实现成为可能，人工智能技术在土木工程结构检测中的应用具有广阔的发展空间。

结 论

本文提出了两种基于人工智能的古建筑表层损伤检测技术。一是以深度卷积神经网络为核心，通过网络的训练得到能够识别三类损伤的分类器，并借助于滑动窗口技术实现了损伤的定位；二是基于候选区域目标的深度学习算法，即 Faster-RCNN，以点对点的网络训练形式，实现了结构损伤的准实时识别定位。以北京故宫的城墙结构为主要研究对象，通过一系列的理论分析和实验验证，得到结论如下：

1. 古建筑的修复和维护工作大量展开，检测工作为其中的关键环节，而结构大量的损伤均在表层得以体现，因此表层的损伤检测具有重大的指导意义。
2. 基于深度学习方法的检测技术仅仅需要结构图像的采集，卷积神经网络模型的搭建以及计算机的训练与测试，相比于当前的人工检测方式效率更高，成本更低。
3. 基于深度学习方法的检测技术无需进行人工的特征提取，实验结果表明该方法受到噪声的影响很小，相比于一般基于图像处理技术的机器视觉方法具有更高的准确性与鲁棒性。
4. 以北京故宫的城墙结构为研究对象，将卷积神经网络与滑动窗口算法结合，实现了损伤的识别与定位任务，结果表明该方法具有准确、高效和实用的特点。增大数据样本数量和加深网络均可提高损伤识别的性能，数据量较少的情况下增大数据效果最为明显；数据充足的情况下加深网络效果最为明显。然而滑动窗口扫描策略对于复杂的实际情况，其效率和适应性均具有一定的局限。
5. 为进一步改进滑动窗口算法的局限性，提出了基于候选区域目标的损伤检测技术，并应用 Faster-RCNN 算法成功实现了损伤的直接定位，然而由于样本数量有限，测试精度并不是特别高。
6. 为实现高效的数据样本采集工作，建立“结构损伤大数据”，初步提出了一个众包式损伤检测模式，能够大量减少数据收集的时间，提升效率。

在未来的研究中，将会继续完善本文第五章和第六章的工作。相比于基于滑动窗口算法的深度学习损伤检测，基于候选区域目标的损伤检测具有更好的实际应用前景。通过扩大样本容量以及进一步改进算法模型，来提高损伤检测的精确度。未来将会落实“众包检测”这一模式，逐步建立“结构损伤大数据”。以深度学习为核心的人工智能技术在土木工程领域具有广阔的发展空间。

参 考 文 献

- [1] Claudio Modena, Maria Rosa Valluzzi, Francesca da Porto, et al. Structural Aspects of The Conservation of Historic Masonry Constructions in Seismic Areas: Remedial Measures and Emergency Actions[J]. International Journal of Architectural Heritage, 2010, 5(4-5):539-558.
- [2] 陈蔚. 我国建筑遗产保护理论和方法研究[D]. 重庆大学, 2006.
- [3] Luigia B, Antonella S. Research on historic structures in seismic areas in Italy[J]. Progress in Structural Engineering & Materials, 2005, 7(2):71-85.
- [4] Elert K, Cultrone G, Navarro C R, et al. Durability of bricks used in the conservation of historic buildings — influence of composition and microstructure[J]. Journal of Cultural Heritage, 2003, 4(2):91-99.
- [5] 赵鹏. 荷载与环境作用下青砖及其砌体结构的损伤劣化规律与机理[D]. 东南大学, 2015.
- [6] 国家文物局. 国家文物事业发展“十三五”规划[EB/OL]. [2017, 04, 21].
http://www.sach.gov.cn/art/2017/2/21/art_1030_137374.html
- [7] 程锡锋, 王金满. 徽州区实施古建筑保护利用工程[EB/OL]. [2017, 04, 21].
<http://www.newshs.com/a/20140213/00180.htm>
- [8] 田牛强. 铜仁市投入 6000 万元保护性维修古建筑群[EB/OL]. [2017, 04, 21].
<http://www.tongrenshi.com/show-9-9923-1.html>
- [9] 刘海玮. 珠海高新区:增加投入修缮古建筑并活化利用[J]. 城乡建设, 2016(9):42-42.
- [10] 张碧. 国家文物局投资 2610 万保护永乐宫古建筑[EB/OL]. [2017, 04, 21].
<http://www.chinanews.com/cul/2014/03-26/5993746.shtml>
- [11] Zhang S, Lu R. ICA 3 D – Intelligent computer-aided ancient Chinese architecture design[J]. Advanced Engineering Informatics, 2012, 26(4):705-715.
- [12] Fang D P, Iwasaki S, Yu M H, et al. Ancient Chinese Timber Architecture. I: Experimental Study[J]. Journal of Structural Engineering, 2001, 127(11):1348-1357.
- [13] 侯卫东. 中国古代砖石建筑及其保护修复概述[J]. 中国文物科学研究, 2012(2):50-53.
- [14] 张帆. 近代历史建筑保护修复技术与评价研究[D]. 天津大学, 2010.
- [15] 中华人民共和国建设部. GB/T 50344-2004 建筑结构检测技术标准[S]. 北京: 中国建筑工业出版社, 2004.
- [16] Balageas D. Introduction to structural health monitoring[M]. ISTE, 2006.
- [17] 丁绍祥. 砌体结构加固工程技术手册[M]. 武汉: 华中科技大学出版社, 2008.
- [18] Carino N J. Nondestructive test methods[J]. Concrete Construction Engineering Handbook, 1997.
- [19] Kim S, Pakzad S, Culler D, et al. Health monitoring of civil infrastructures using wireless sensor networks[C]//Proceedings of the 6th international conference on Information processing in sensor networks. ACM, 2007: 254-263.
- [20] Lovse J W, Teskey W F, Lachapelle G, et al. Dynamic Deformation Monitoring of Tall Structure Using GPS Technology[J]. Journal of Surveying Engineering, 1995, 121(1):35-40.

- [21] Glisic B, Inaudi D, Nan C. Pile Monitoring with Fiber Optic Sensors During Axial Compression, Pullout, and Flexure Tests[J]. Transportation Research Record Journal of the Transportation Research Board, 2002, 1808(1):11-20.
- [22] Ramos T, Furtado A, Eslami S, et al. 2D and 3D Digital Image Correlation in Civil Engineering—Measurements in a Masonry Wall[J]. Procedia Engineering, 2015, 114: 215-222.
- [23] Ferrer A D D, Escalante E E, Barbosa S R J, et al. Material deformation estimation with Computer Vision methods[C]//Signal Processing, Images and Computer Vision (STSIVA), 2015 20th Symposium on. IEEE, 2015: 1-5.
- [24] Rajaram S, Vanniamparambil P A, Khan F, et al. Full - field deformation measurements during seismic loading of masonry buildings[J]. Structural Control and Health Monitoring, 2016.
- [25] Pan B, Qian K, Xie H, et al. Two-dimensional digital image correlation for in-plane displacement and strain measurement: a review[J]. Measurement science and technology, 2009, 20(6): 062001.
- [26] Xu F, Li B, Wang K. Study and Application of Displacement Back Analysis Based on CACA-SVM[C]//Applied Mechanics and Materials. Trans Tech Publications, 2013, 353: 142-145.
- [27] Li X, Wang F, Cai Y. Predicting Deformations of Tunnel Surrounding Rock by Using Least Squares Support Vector Machine[J]. Journal of Highway & Transportation Research & Development, 2011, 5(1):45-50.
- [28] 李勤, 盛金喜, 田飞, 等. 遗产保护中砌体古建筑的检测与鉴定研究[C]// 全国工程结构安全检测鉴定与加固修复研讨会. 2015.
- [29] 中华人民共和国建设部. GB 50203-2011 砌体工程施工质量验收规范[S]. 北京: 中国建筑工业出版社, 2011.
- [30] 何流. “酥碱”的词源探究及现代学理浅析[J]. 中国科技语, 2012, 14(6):47-49.
- [31] Kabir S, Rivard P, He D C, et al. Damage assessment for concrete structure using image processing techniques on acoustic borehole imagery[J]. Construction & Building Materials, 2009, 23(10):3166-3174.
- [32] Canny J. A computational approach to edge detection[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 1986, 8(6):679.
- [33] Antonini M, Barlaud M, Mathieu P, et al. Image coding using wavelet transform[J]. IEEE Transactions on Image Processing, 1992, 1(1):205-220.
- [34] Haralick R M. Statistical and structural approaches to texture[J]. Proceedings of the IEEE, 1979, 67(5): 786-804.
- [35] 段瑞玲, 李庆祥, 李玉和. 图像边缘检测方法研究综述[J]. 光学技术, 2005, 31(3):415-419.
- [36] 管宏蕊, 丁辉. 图像边缘检测经典算法研究综述[J]. 首都师范大学学报: 自然科学版, 2009 (S1): 66-69.
- [37] Chien Y. Pattern classification and scene analysis[J]. The Library Quarterly: Information, Community, Policy, 1974, 19(Volume 44, Number 3):462-463.
- [38] Tippett, James T. Optical and electro-optical information processing[M]. Massachusetts Institute of Technology Press, 1965.
- [39] Gonzalez R C, Woods R E. Image processing[J]. Digital image processing, 2007, 2.

- [40]徐辛超, 付晨, 徐爱功. 一种改进的 Canny 边缘提取方法[J]. 遥感信息, 2016, 31(5):43-46.
- [41]Abdel-Qader I, Abudayyeh O, Kelly M E. Analysis of edge-detection techniques for crack identification in bridges[J]. Journal of Computing in Civil Engineering, 2003, 17(4): 255-263.
- [42]Corr D, Accardi M, Graham-Brady L, et al. Digital image correlation analysis of interfacial debonding properties and fracture behavior in concrete[J]. Engineering Fracture Mechanics, 2007, 74(1): 109-121.
- [43]Falsone G, Lombardo M. Stochastic representation of the mechanical properties of irregular masonry structures[J]. International Journal of Solids and Structures, 2007, 44(25): 8600-8612.
- [44]Ghiassi B, Xavier J, Oliveira D V, et al. Application of digital image correlation in investigating the bond between FRP and masonry[J]. Composite Structures, 2013, 106: 340-349.
- [45]Gencturk B, Hossain K, Kapadia A, et al. Use of digital image correlation technique in full-scale testing of prestressed concrete structures[J]. Measurement, 2014, 47: 505-515.
- [46]Cavalagli N, Cluni F, Gusella V. Evaluation of a Statistically Equivalent Periodic Unit Cell for a quasi-periodic masonry[J]. International Journal of Solids and Structures, 2013, 50(25): 4226-4240.
- [47]Nejad F M, Motekhases F Z, Zakeri H, et al. An Image Processing Approach to Asphalt Concrete Feature Extraction[J]. Journal of Industrial and Intelligent Information Vol, 2015, 3(1).
- [48]Mahal M, Blanksvärd T, Täljsten B, et al. Using digital image correlation to evaluate fatigue behavior of strengthened reinforced concrete beams[J]. Engineering Structures, 2015, 105: 277-288.
- [49]Hamrat M, Boulekbache B, Chemrouk M, et al. Flexural cracking behavior of normal strength, high strength and high strength fiber concrete beams, using Digital Image Correlation technique[J]. Construction and Building Materials, 2016, 106: 678-692.
- [50]Harrington P. Machine learning in action[M]. Greenwich, CT: Manning, 2012.
- [51]Liu S W, Huang J H, Sung J C, et al. Detection of cracks using neural networks and computational mechanics[J]. Computer methods in applied mechanics and engineering, 2002, 191(25): 2831-2845.
- [52]Jiang X, Adeli H. Pseudospectra, MUSIC, and dynamic wavelet neural network for damage detection of highrise buildings[J]. International Journal for Numerical Methods in Engineering, 2007, 71(5): 606-629.
- [53]Sipos T K, Sigmund V, Hadzima-Nyarko M. Earthquake performance of infilled frames using neural networks and experimental database[J]. Engineering structures, 2013, 51: 113-127.
- [54]Butcher J B, Day C R, Austin J C, et al. Defect detection in reinforced concrete using random neural architectures[J]. Computer-Aided Civil and Infrastructure Engineering, 2014, 29(3): 191-207.
- [55]Kabir S. Imaging-based detection of AAR induced map-crack damage in concrete structure[J]. NDT & E International, 2010, 43(6): 461-469.
- [56]Moon H, Kim J. Intelligent crack detecting algorithm on the concrete crack image using neural network[J]. Proceedings of the 28th ISARC, 2011: 1461-1467.
- [57]Jahanshahi M R, Masri S F, Padgett C W, et al. An innovative methodology for detection and quantification of cracks through incorporation of depth perception[J]. Machine vision and applications, 2013, 24(2): 227-241.

- [58] O' Byrne M, Schoefs F, Ghosh B, et al. Texture analysis based damage detection of ageing infrastructural elements[J]. Computer-Aided Civil and Infrastructure Engineering, 2013, 28(3): 162-177.
- [59] Plevris V, Asteris P G. Modeling of masonry failure surface under biaxial compressive stress using Neural Networks[J]. Construction and Building Materials, 2014, 55: 447-461.
- [60] O' Byrne M, Ghosh B, Schoefs F, et al. Regionally enhanced multiphase segmentation technique for damaged surfaces[J]. Computer-Aided Civil and Infrastructure Engineering, 2014, 29(9): 644-658.
- [61] Wu L, Mokhtari S, Nazef A, et al. Improvement of crack-detection accuracy using a novel crack defragmentation technique in image-based road assessment[J]. Journal of Computing in Civil Engineering, 2014, 30(1): 04014118.
- [62] Hinton G E, Osindero S, Teh Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7):1527-1554.
- [63] 张重生. 深度学习原理与应用实践[M]. 北京: 电子工业出版社, 2016.
- [64] Hubel D H, Wiesel T N. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex.[J]. Journal of Physiology, 1962, 160(1):106.
- [65] Rumelhart D E, Hinton G E, Williams R J. Learning rep-resentation by back-propagating errors[J]. Nature, 1986, 323(3):533-536.
- [66] Lecun Y, Boser B, Denker J S, et al. Backpropagation Applied to Handwritten Zip Code Recognition[J]. Neural Computation, 1989, 1(4):541-551.
- [67] 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.
- [68] Stanford Vision Lab. Large Scale Visual Recognition Challenge (ILSVRC)[EB/OL]. [2017, 04, 24].
<http://www.image-net.org/challenges/LSVRC/>
- [69] Stanford Vision Lab. ImageNet.[DB/OL]. [2017, 04, 24]. <http://www.image-net.org/>
- [70] Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks[C]// International Conference on Neural Information Processing Systems. Curran Associates Inc. 2012:1097-1105.
- [71] Zeiler M D, Fergus R. Visualizing and understanding convolutional networks[C]//European conference on computer vision. Springer International Publishing, 2014: 818-833.
- [72] 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.
- [73] Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition[J]. arXiv preprint arXiv:1409.1556, 2014.
- [74] He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition[C]// Computer Vision and Pattern Recognition. IEEE, 2016:770-778.
- [75] Szegedy C, Ioffe S, Vanhoucke V, et al. Inception-v4, inception-resnet and the impact of residual connections on learning[J]. arXiv preprint arXiv:1602.07261, 2016.
- [76] Oullette R, Browne M, Hirasawa K. Genetic algorithm optimization of a convolutional neural network for autonomous crack detection[C]//Evolutionary Computation, 2004. CEC2004. Congress on. IEEE, 2004, 1: 516-521.

- [77] Makantasis K, Protopapadakis E, Doulamis A, et al. Deep convolutional neural networks for efficient vision based tunnel inspection[C]//Intelligent Computer Communication and Processing (ICCP), 2015 IEEE International Conference on. IEEE, 2015: 335-342.
- [78] Protopapadakis E, Makantasis K, Kopsiaftis G, et al. Crack identification via user feedback, convolutional neural networks and laser scanners for tunnel infrastructures[C]//the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications, 2016: 725-734.
- [79] Abdeljaber O, Avci O, Kiranyaz S, et al. Real-time vibration-based structural damage detection using one-dimensional convolutional neural networks[J]. Journal of Sound and Vibration, 2017, 388: 154-170.
- [80] Cha Y J, Choi W, Buyukozturk O. Deep learning-based crack damage detection using convolutional neural network"[J]. Computer-Aided Civil and Infrastructure Engineering, 2017, 32(3): 2013-2014.
- [81] Cover T M, Hart P E. Nearest neighbor pattern classification[J]. IEEE Transactions on Information Theory, 1967, 13(1):21-27.
- [82] Cristianini N, Shawetaylor J. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods: Notation[J]. 2000, 32(8):1-28.
- [83] Fisher R A. The use of multiple measurements in taxonomic problems[J]. Annals of eugenics, 1936, 7(2): 179-188.
- [84] Hsu C W, Lin C J. A comparison of methods for multiclass support vector machines[J]. IEEE transactions on Neural Networks, 2002, 13(2): 415-425.
- [85] Bishop C M. Pattern recognition[J]. Machine Learning, 2006, 128: 1-58.
- [86] Anguita D, Ridella S, Rivieccio F, et al. Hyperparameter design criteria for support vector classifiers[J]. Neurocomputing, 2003, 55(1–2):109-134.
- [87] Chang C C, Lin C J. LIBSVM: A library for support vector machines[M]. ACM, 2011.
- [88] Hawkins D M. The Problem of Overfitting[J]. Journal of Chemical Information & Computer Sciences, 2004, 44(1):1.
- [89] Boer P T D, Kroese D P, Mannor S, et al. A Tutorial on the Cross-Entropy Method[J]. Annals of Operations Research, 2005, 134(1):19-67.
- [90] Nair V, Hinton G E. Rectified Linear Units Improve Restricted Boltzmann Machines[C]// International Conference on Machine Learning. DBLP, 2010:807-814.
- [91] Maas A L, Hannun A Y, Ng A Y. Rectifier nonlinearities improve neural network acoustic models[C]//Proc. ICML. 2013, 30(1).
- [92] 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.
- [93] Xu B, Wang N, Chen T, et al. Empirical evaluation of rectified activations in convolutional network[J]. arXiv preprint arXiv:1505.00853, 2015.
- [94] Cybenko G. Approximation by superpositions of a sigmoidal function[J]. Mathematics of Control, Signals, and Systems (MCSS), 1989, 2(4): 303-314.

- [95] Scherer D, Müller A, Behnke S. Evaluation of pooling operations in convolutional architectures for object recognition[J]. Artificial Neural Networks—ICANN 2010, 2010: 92-101.
- [96] Glorot X, Bengio Y. Understanding the difficulty of training deep feedforward neural networks[C]//Aistats. 2010, 9: 249-256.
- [97] Qian N. On the momentum term in gradient descent learning algorithms[J]. Neural networks, 1999, 12(1): 145-151.
- [98] Zeiler M D. ADADELTA: an adaptive learning rate method[J]. arXiv preprint arXiv:1212.5701, 2012.
- [99] Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization[J]. Journal of Machine Learning Research, 2011, 12(Jul): 2121-2159.
- [100] Kingma D, Ba J. Adam: A method for stochastic optimization[J]. arXiv preprint arXiv:1412.6980, 2014.
- [101] Nesterov Y. A method for unconstrained convex minimization problem with the rate of convergence $O(1/k^2)$ [C]//Doklady an SSSR. 1983, 269(3): 543-547.
- [102] Geoff Hinton. Lecture 6a: Overview of mini-batch gradient descent[EB/OL]. [2017, 05, 02].
http://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf
- [103] Simard P Y, Steinkraus D, Platt J C. Best practices for convolutional neural networks applied to visual document analysis[C]// International Conference on Document Analysis and Recognition, 2003. Proceedings. IEEE, 2003:958.
- [104] Cireşan D C, Meier U, Gambardella L M, et al. Deep, big, simple neural nets for handwritten digit recognition[J]. Neural computation, 2010, 22(12): 3207-3220.
- [105] Chawla N V, Bowyer K W, Hall L O, et al. SMOTE: synthetic minority over-sampling technique[J]. Journal of Artificial Intelligence Research, 2011, 16(1):321-357.
- [106] Lindholm E, Nickolls J, Oberman S, et al. NVIDIA Tesla: A Unified Graphics and Computing Architecture[J]. IEEE Micro, 2008, 28(2):39-55.
- [107] NVIDIA. What is GPU-accelerated computing?[EB/OL]. [2017, 05, 08].
<http://www.nvidia.com/object/what-is-gpu-computing.html>
- [108] Nickolls J, Dally W J. The GPU computing era[J]. Micro IEEE, 2010, 30(2):56-69.
- [109] NVIDIA. CUDA Parallel Computing Platform[EB/OL]. [2017, 05, 08].
http://www.nvidia.com/object/cuda_home_new.html
- [110] GitHub. BVLC/Caffe[CP/OL]. [2017, 05, 08]. <https://github.com/BVLC/caffe/tree/windows>
- [111] Jia Y, Shelhamer E. Caffe[EB/OL]. [2017, 05, 08]. <http://caffe.berkeleyvision.org/>
- [112] 乐毅, 王斌. 深度学习——Caffe 之经典模型详解与实战[M]. 北京: 电子工业出版社, 2016.
- [113] 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.
- [114] Uijlings J R R, Van De Sande K E A, Gevers T, et al. Selective search for object recognition[J]. International journal of computer vision, 2013, 104(2): 154-171.
- [115] Felzenszwalb P F, Huttenlocher D P. Efficient graph-based image segmentation[J]. International journal of computer vision, 2004, 59(2): 167-181.

- [116] He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[C]/European Conference on Computer Vision. Springer International Publishing, 2014: 346-361.
- [117] Girshick R. Fast r-cnn[C]/Proceedings of the IEEE International Conference on Computer Vision. 2015: 1440-1448.
- [118] Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]/Advances in neural information processing systems. 2015: 91-99.
- [119] GitHub. ShaoqingRen/faster_rcnn[CP/OL]. [2017, 05, 18].
https://github.com/ShaoqingRen/faster_rcnn
- [120] Everingham M, Van Gool L, Williams C K I, et al. The PASCAL Visual Object Classes Homepage [EB/OL]. [2017, 05, 18]. <http://host.robots.ox.ac.uk/pascal/VOC/>
- [121] Everingham M, Van Gool L, Williams C K I, et al. The pascal visual object classes (voc) challenge[J]. International journal of computer vision, 2010, 88(2): 303-338.
- [122] GitHub. tzutalin/labelImg[CP/OL]. [2017, 05, 18]. <https://github.com/tzutalin/labelImg>
- [123] Howe J. CROWDSOURCING: A DEFINITION[EB/OL]. [2017, 05, 27].
<http://crowdsourcing.typepad.com/cs/2006/06/>
- [124] 百度百科. 众包[EB/OL]. [2017, 04, 21].
<http://baike.baidu.com/item/%E4%BC%97%E5%8C%85/548975>
- [125] Zhao X, Han R, Ding Y, et al. Portable and convenient cable force measurement using smartphone[J]. Journal of Civil Structural Health Monitoring, 2015, 5(4): 481-491.
- [126] Zhao X, Zhao Q, Yu Y, et al. Distributed Displacement Response Investigation Technique for Bridge Structures Using Smartphones[J]. Journal of Performance of Constructed Facilities, 2017, 31(4): 04017029.

附录 攻读本科学位期间发表学术论文情况

- 1 Distributed Displacement Response Investigation Technique for Bridge Structures Using Smartphones. Xuefeng Zhao, **Qingan Zhao**, Yan Yu, Yuting Chen, Hao Liu, Mingchu Li, Jinping Ou. Journal of Performance of Constructed Facilities, 2017, 31(4): 04017029. **SCI 检索** (影响因子 0.893) , DOI: 10.1061/(ASCE)CF.1943-5509.0001025
- 2 智能手机结构位移监测新技术. 赵雪峰, **赵庆安**, 喻言, 刘昊, 李明楚, 欧进萍. 物联网技术, 2016, 6(4):13-17.
- 3 Smartphone Based Public Participant Emergency Rescue Information Platform for Earthquake Zone - “E-Explorer”. Deli Peng, Xuefeng Zhao, **Qingan Zhao**, Yan Yu. Vibroengineering Procedia, 2015, 5:436-439. **EI 检索**, 检索号: 20160301821450
- 4 Vision-based Damage Identification and Assessment for Masonry Historic Structures Using Convolutional Neural Networks. **Qingan Zhao**, Peng Zhao, Xuefeng Zhao. Preventive Conservation in Historic Houses and Palace-Museums: Assessment Methodologies and Application. (本毕业论文第三、四章) (已录用)
- 5 Damage Detection for Masonry Historic Structures Based on Deep Learning. **Qingan Zhao**, Peng Zhao, Xuefeng Zhao. Computer-Aided Civil and Infrastructure Engineering. **SCI 检索** (影响因子 5.288) . (本毕业论文第三、四、五章) (待投稿)

致 谢

本文研究从题目方向的确定至最终论文的完成，全部由导师赵雪峰教授悉心指导；所需要的全部研究和实验设备等硬件条件，也全部由赵老师提供。从 2014 年 9 月加入学部的抗震创新实验班，我便在赵老师的指导下开始了本科期间的科研工作。三年来，从最开始做的一些基础工作，进入实验室帮忙，到在赵老师的指导下代表学部、学校参加各级科创比赛并获奖，自主设计完成实验并发表第一篇 SCI 论文，再到开展自己的独立研究，完成毕业论文，在赵老师的大力帮助下成功申请到美国加州大学伯克利分校攻读研究生。在这三年中赵老师尽心尽力的培养我历历在目，终生难忘。赵老师用言传身教的方式向我诠释了“创新”和“努力”这两个在科研中最重要的品质，使我无论在科研、学习还是生活中都能够勇于克服困难，另辟蹊径。同时，赵老师不仅是我学术上的楷模，更在平时的交流中深深影响着我的做人和生活方式，常常推荐我阅读一些能够扩展视野、陶冶情操的书籍，并训练我很多科研以外的优秀能力。在此对赵老师表示崇高的敬意和深深的感谢！

感谢北京故宫博物院古建部赵鹏副主任在本文古建筑损伤情况及分类等方面对我的指导，以及提供本文实验所用到的城墙图像样本；感谢李生元师兄在选题和后续的技术实现和编程方面对我的帮助；感谢方腾伟师弟、赵鑫如师妹在人工分割训练样本时所作的贡献；感谢国家自然科学基金面上项目（51479031、51278085）对本文研究的支持。

感谢欧进萍院士为智能结构研究所创造的科研环境，感谢韩瑞聪师姐和已经毕业的刘昊师兄、彭德利师兄帮助引领我进入科研的道路。感谢同师门的张阳、苏红果、李光、王念念、李金珂师兄师姐对我在科研和生活上的帮助，感谢完成毕业论文期间在 309 教研室的师兄师姐给予的帮助与照顾。

感谢班主任崔瑶副教授四年来的巨大帮助和鼓励，从大学一年级为我指引努力的方向一直到大学四年级帮助我成功申请到梦校，在此向崔老师表示深深的感谢！

感谢所有本科期间帮助和教导过我的老师；感谢辅导员田苗老师的帮助与照顾；感谢学校和学部提供的资源和平台；感谢土木 1301 班大家庭，让我在这四年学习和生活中体会到了如兄弟姐妹般的温暖；感谢“伟大的 231 寝室”室友张惟一、冷慧康、李壮壮，四年来的相互帮助，一起学习和游戏的时光我将永远不会忘记，愿大家前程似锦！

特别感谢家人的帮助与支持，没有父母的养育之恩就没有我的今天，大爱无以为报，唯有不断奋斗与进取，让自己更加优秀！在赴美留学期间为校争光，为国争光！

赵庆安

2017.05.30