武汉理工大学毕业设计（论文）

基于机器视觉的目标识别与分类

学院（系） 信息工程学院

专业班级：电信工程1501班

学生姓名： 吴博

指导教师： 肖攸安 教授

学位论文原创性声明

本人郑重声明：所呈交的论文是本人在导师的指导下独立进行研究所取得的研究成果。除了文中特别加以标注引用的内容外，本论文不包括任何其他个人或集体已经发表或撰写的成果作品。本人完全意识到本声明的法律后果由本人承担。

作者签名：

年 月 日

学位论文版权使用授权书

本学位论文作者完全了解学校有关保障、使用学位论文的规定，同意学校保留并向有关学位论文管理部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权省级优秀学士论文评选机构将本学位论文的全部或部分内容编入有关数据进行检索，可以采用影印、缩印或扫描等复制手段保存和汇编本学位论文。

本学位论文属于1、保密囗，在 年解密后适用本授权书

2、不保密囗 。

（请在以上相应方框内打“√”）

作者签名： 年 月 日

导师签名： 年 月 日

摘要

本文使用Mask Rcnn算法，针对金属腐蚀缺陷检测的应用需求，基于Python + Opencv环境，开发了一个针对金属腐蚀图片的腐蚀缺陷目标检测与分类系统。

由于传统的腐蚀缺陷检测方法一般都是通过腐蚀部位取样以及化学分析，电子显微镜获得腐蚀信息，但这些检测方法仅限于实验室分析，有的方法会对检测目标造成损伤，难用于现场检测。而基于机器视觉的检测方法是通过采集目标图像并对图像进行分析得出腐蚀信息，可以避开这些问题。

论文以标记过的金属腐蚀图片为数据集进行有监督训练，建立了金属腐蚀模型对金属腐蚀图像进行预测，并将检测到的金属腐蚀缺陷以可视化的形式标记在目标图像中。

结果表明，在对验证集的预测中，建立的腐蚀模型可以较好的识别点腐蚀和缝隙腐蚀。

**Abstract**

In this paper, using the Mask Rcnn algorithm, based on the application requirements of metal corrosion defect detection, based on the Python + Opencv environment, a corrosion defect target detection and classification system for metal corrosion pictures was developed.

Since the traditional corrosion defect detection methods generally use corrosion sampling and chemical analysis, electron microscopy obtains corrosion information, but these detection methods are limited to laboratory analysis, and some methods may cause damage to the detection target and are difficult to be used for on-site inspection.

The machine vision-based detection method avoids these problems by collecting the target image and analyzing the image to obtain corrosion information.

The paper uses the marked metal corrosion picture as the data set for supervised training, establishes a metal corrosion model to predict the metal corrosion image, and marks the detected metal corrosion defects in the target image in a visual form.

The results show that the corrosion model established can better identify pitting corrosion and crevice corrosion in the prediction of the verification set.

**Key Words:** Mask Rcnn; machine vision;metal corrosion detection

目录

[第1章 绪论 1](#_Toc9947969)

[1.1目的及意义 1](#_Toc9947970)

[1.2国内外研究现状 1](#_Toc9947971)

[1.3方案选择 3](#_Toc9947972)

[1.4论文整体结构 3](#_Toc9947973)

[第2章 系统结构与预期 5](#_Toc9947974)

[2.1系统结构 5](#_Toc9947975)

[2.2系统预期功能与指标 6](#_Toc9947976)

[第3章 Mask R-CNN介绍 7](#_Toc9947977)

[第4章 模型训练 14](#_Toc9947978)

[4.1环境配置 14](#_Toc9947979)

[4.2数据集制作 15](#_Toc9947980)

[4.3模型训练 17](#_Toc9947981)

[第5章 UI界面设计 20](#_Toc9947982)

[5.1GUI工具选择 20](#_Toc9947983)

[5.2环境配置 21](#_Toc9947984)

[5.3窗口设计 21](#_Toc9947985)

[第6章 系统测试 26](#_Toc9947986)

[6.1系统功能测试 26](#_Toc9947987)

[6.2系统指标测量 28](#_Toc9947988)

[第7章 总结 29](#_Toc9947989)

[参考文献 30](#_Toc9947990)

[第8章 致谢 32](#_Toc9947991)

# 绪论

## 1.1目的及意义

本毕业设计的课题是基于机器视觉的目标检测与分类。自进入工业化社会以来，制造业直接体现着一个国家的生产水平。随着技术领先国家一些战略的相继出台，依托网络信息技术、大数据、云计算等技术的深度融合与集成，传统制造业正逐步迈入新时代，只有不断提升制造业自动化与智能化水平才能加快我国制造业的转型升级。作为近年来自动化领域迅猛发展的高智能化技术代表——机器视觉技术，凭借其可视化优势，迅速跻身于制造业自动化的关键位置，从而极大地推动了工业发展的智能化进程。人类将自身视觉系统的原理移植于视觉检测技术中，发明了视觉传感器与机器视觉系统，并广泛应用于工业、农业、医药、军事、航天、交通、安全、科研等领域，取得了巨大的经济与社会效益，机器视觉市场潜力巨大。

随着社会步入智能化时代，人工智能[1-2]的发展也日益火热。机器视觉技术就是融合人工智能的一项技术，通常指利用计算机通过视觉的角度模拟生物的视觉感知功能，即对人眼的感知进行延伸与扩充，其功能主要是从观察到的事物图像中提取空间信息，加以处理与理解。

我国在材料腐蚀领域，为监测腐蚀数据进行的试验方法相对单一，大多采取人工的方式，通过肉眼观察相应的材料外观腐蚀图像，来获取材料的腐蚀特征相关数据信息。

金属材料由于各种优良性能，在工业及生活中得到了广泛应用。但应用中的金属材料常暴露在自然环境及与酸，碱，盐等液体接触，很容易发生腐蚀。工业过程中，普遍存在的金属材料腐蚀会给工业带来巨大损失。传统的腐蚀检测方法具有检测慢，准确率低，条件苛刻，成本高等多种特点， 所以使用机器视觉技术进行腐蚀检测具有重大意义。

## 1.2国内外研究现状

国内发展现状：对于金属腐蚀形貌特征的提取和处理和腐蚀图像处理方面，大连理工大学张纬进行了金属腐蚀形貌特征提取用于腐蚀诊断[3]的研究，探讨通过研究金属表面腐蚀图像来计算腐蚀孔面积和腐蚀孔数目的方法,对腐蚀图像进行量化描述。罗晓琳研究了基于图像的金属腐蚀区域分割[4]问题，用像素差异的变化进行腐蚀区域的分割，为金属腐蚀区检测提供了参考。李忠虎，张琳等研究了一种基于BP神经网络的图像边缘检测算法，用于管道腐蚀检测，与传统的边缘检测算法检测结果进行比对，明显提高了检测精度及抗噪声能力。重庆理工大学李丙涛提出了一种腐蚀特征图像分析系统[5]，减少了人为因素的影响，更适合定量化处理。熊岩提出了构建基于数字图像处理技术的金属板材腐蚀实验自动评价系统[6]的方案，并对腐蚀区域的提取技术做了重点研究。陶蕾将数字图像处理技术以及电化学测试技术应用于自然环境材料腐蚀检测研究中，建立了适用于实验室和大气环境的腐蚀形貌图像处理系统[7]。贺东海借助数字图像处理技术实现对镁合金腐蚀过程和程度以及防护膜形成过程的分析评价，建立起一种不同于传统研究方法的镁合金腐蚀和防护膜图像识别技术[8]。纪钢等在RGB空间获取材料腐蚀特征图像信息，使用 HSV颜色空间描述腐蚀特征区域的颜色值，完成了在HSV下的材料腐蚀特征区域颜色与边界量化处理[9]，降低了后期腐蚀等级评价时的误差，提高了腐蚀等级评价的准确度。在研究完整的金属腐蚀检测系统方面，纪钢、朱烨华等设计了基于计算机视觉的材料外观腐蚀特征在线检测系统[10]，对金属腐蚀进行分类识别。于淼提出了一个利用镀层腐蚀图像信息分析来进行自动评级的系统[11]，利用腐蚀图像特征点的形态与腐蚀图像失光、变色信息进行相关的处理,从而识别出缺陷图像的腐蚀等级。

国外发展现状：A.Stankiewicz等使用机器视觉法评价镍-磷-纳米氧化锆复合涂层的耐腐蚀性[12]，用机器视觉法测定腐蚀产物样品表面覆盖度。Alicja Stankiewicz等设计出一个用于腐蚀检测的机器视觉系统[13]，确定涂层表面的腐蚀程度。Rogelio Ramos等开发了一个基于LabVIEW2010计算机视觉平台的虚拟仪器[14]，应用于点蚀研究中。Suchart Yammen等设计了一种先进的硬盘磁极尖端腐蚀自动检测视觉系统[15]，能够检测用铁膜制作的极尖关键部位的腐蚀情况。Luca Petricca等对金属腐蚀自动检测的标准计算机视觉技术和深度学习方法[16]进行了比较，通过对图像进行分类，计算两种不同方法的总精度，进行了测试。B.B.Zaidan等设计一种检测金属内部腐蚀的方法[17]，在纹理分析的基础上，该方法的主要功能是边缘检测、结构元素和图像扩展，通过对图像中腐蚀部分的检测，取得了较好的效果。Deegan J Atha等提出了不同的基于卷积神经网络的金属表面腐蚀评估方法[18]，研究表明，与最先进的预训练卷积神经网络相比，所提出的卷积神经网络在保持腐蚀检测的可比性能的同时，显著提高了计算时间。Nadiah Amalina Zulkiﬂi等提出了一种基于图像的锅炉集箱内部裂纹和腐蚀检测方法[19]，对锅炉集箱内壁图像进行采集后，应用阈值技术对采集图像的背景变化进行管理，然后利用边缘检测算法对腐蚀区边界进行识别。M Enikeev提出一种解决铝表面点蚀识别问题的机器学习方法[20]，研究结果可在可视化分析的基础上用于腐蚀过程及其机理的研究。

总体来说，国内对金属腐蚀的相关研究使用传统的步骤：图像预处理，图像分割，图像特征提取，图像分类等进行图像检测，研究对金属腐蚀的特征提取，图像分割，特征参数测量与分析等各方面的方法。相比较而言，国外对金属腐蚀的研究则较多的使用神经网络对图像进行检测和识别。由于要设置图像分割的阈值，选择用于识别和训练的缺陷特征，传统的图像检测方法需要对金属腐蚀的特征非常了解才能进行，而使用神经网络以图片集为数据集进行训练生成预测模型，再对图像进行预测则避开了这些问题。

## 1.3方案选择

若选择图像预处理，图像分割，缺陷特征提取，缺陷识别与分类四个步骤的传统方案，关键问题在于一幅金属腐蚀图像中存在多种腐蚀类别，无法直接对单重腐蚀缺陷进行特征提取，需要首先对图像进行腐蚀分区，使得一块区域中只有一种类别的腐蚀然后分别进行检测。这就需要用到区域分割方法，但由于许多金属腐蚀缺陷没有明显边界，寻找一种较好区域分割方法较困难。为了避开区域分割这个问题，本文可以使用基于候选区域的目标检测算法。

深度学习在解决图像目标识别等方面都表现出色，在各种神经网络中卷积神经网络（CNN）是被研究的最多的。在计算能力尚不发达，缺乏训练数据的时候，训练出好的卷积神经网络非常困难，但随着计算机的计算能力大幅度提升和类似ImageNet的大规模标记数据的出现，卷积神经网络近年来也开始了快速发展。CNN在计算机视觉方面可以完成许多任务：图像分类，物体检测，视觉显著检测，场景标签等等。自从Ross Girshick提出了Region CNN(RCNN)，已经有许多人在它的基础上做出了进一步的的研究。从R-CNN到Mask R-CNN都是基于候选区域的目标检测算法，可以避开区域分割这个问题，考虑到将金属腐蚀缺陷可视化显示是一个实例分割任务，再综合速度和识别准确率等方面，决定采用Mask R-CNN算法。

## 1.4论文整体结构

本文分为七章，每章的结构和内容如下：

第一章：阐述基于机器视觉的金属腐蚀缺陷检测与分类的目的及意义，在当前领域有何作用。调查国内外近几年在该领域的进展和成果，并分析有何优点和不足。阐述完成毕业设计的技术路线，选择一个可行的方案。

第二章：讲解系统整体的结构和各部分的结构，说明系统应该实现的功能。

第三章：对使用的Mask R-CNN算法进行介绍，主要从结构上讲解各个模块的功能，主要的模块有ResNet101，RPN，anchor，RoIAlign，classifier，bounding box regression。

第四章：算法代码的语言是python，首先对python的安装，各种需要的python包和其他需要用到的前置文件作出了说明，给出了配置环境的步骤。然后说明了图像标注工具labelme的安装以及如何使用它进行图像的标记再转化为数据集。最后说明了如何使用数据集训练模型，以及用训练好的模型预测图片中存在的金属腐蚀缺陷。

第五章：介绍了几种使用python语言开发UI界面的GUI工具，并对它们各自的优缺点进行了分析，从而选出合适的GUI工具，之后对选择的GUI的工具的功能和使用方法作了说明。详细列出了UI界面所有的功能，结合代码讲解了各功能的实现方法并提出了期间遇到的问题和解决办法，最后分析了UI界面中所有控件，信号，槽函数之间的连接关系。

第六章：对整个系统进行测试，包括检测整个系统是否运行正常并显示出结果，对系统操作界面是否会有延迟和卡顿，检测图片的速度如何，输出的结果是否能够正确地返回金属腐蚀缺陷的位置和类别以及对哪些类别的腐蚀缺陷识别的较好，哪些较差，识别率为多少。

第七章：总结全文的工作，分析系统的优缺点并指出影响系统性能指标的各种因素，提出可能的改进方法从而优化系统。

# 系统结构与预期

## 2.1系统结构

从整体来说，金属腐蚀缺陷检测系统主要分为两个部分：算法和模型部分，UI部分。将这两部分按一定结构结合起来即可完成从UI选取图片并进行检测的功能，其中算法和模型部分，UI界面又分为若干小的部分。

算法和模型部分主要目的是训练出可预测金属腐蚀缺陷类别的腐蚀模型，可以对输入的金属腐蚀图片进行预测并得出结果，这个部分也是整个系统的核心部分，腐蚀模型是通过图片集的标注，生成数据集并加以训练得到，算法与模型部分的流程图如图2.1所示。

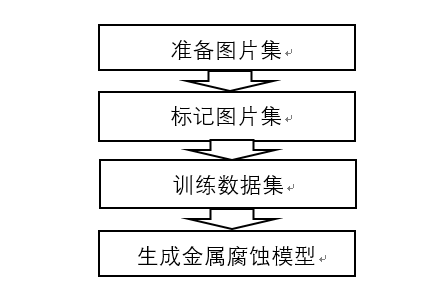


图 2.1 算法与模型部分流程图

UI部分的目的是设计出一个简洁易懂的可视化界面，方便用户在客户端进行金属腐蚀图片的腐蚀种类预测，而不用进入复杂的开发环境中更改代码和运行程序，它主要将预测的结果返回UI，从而使用户能够看到，UI部分的流程图如图2.2所示。

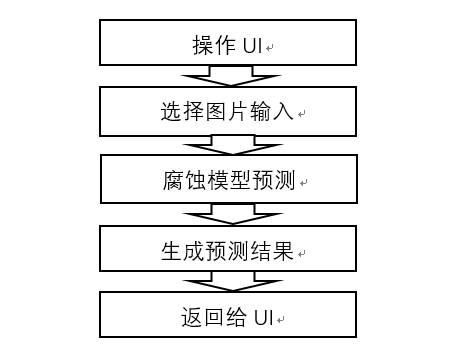


图 2.2 UI部分流程图

## 2.2系统预期功能与指标

系统需要能够让用户在UI界面选取任意的金属腐蚀图片，并通过腐蚀模型预测图片，最后将预测结果以可视化的形式显示在UI界面上反馈给用户。

在算法和腐蚀模型方面，能对训练的四种金属腐蚀缺陷（点腐蚀，缝隙腐蚀，丝状腐蚀，一般攻击腐蚀）进行识别，且识别率达到90%以上。具体来说，腐蚀模型能在要进行预测的目标图片上将腐蚀缺陷的位置标出,并附上标签说明属于哪一种金属腐蚀缺陷。

在UI界面方面，需要给用户提供浏览图片的选项用来选择目标图片，提供测试功能检测与分类目标图片存在的金属腐蚀缺陷，若检测时间较长能有显示检测进度的进度条，优化用户体验，最后能将可视化结果显示在UI界面上返回给用户。

最后需要系统达到一定的指标，期望的各个指标如表2.1所示。

表2.1 系统预期指标

|  |  |
| --- | --- |
| 参数 | 指标 |
| 检测一张图片的时间 | 30s |
| 能识别的腐蚀缺陷种类 | 能识别训练的所有腐蚀缺陷 |
| 单种腐蚀缺陷的识别率 | 90% |

# Mask R-CNN介绍

Mask R-CNN[21]的整体框架如图3.1所示，主要由backbone，FPN，RPN，RoIAlign，anchor，classification，box regression，mask几部分组成。

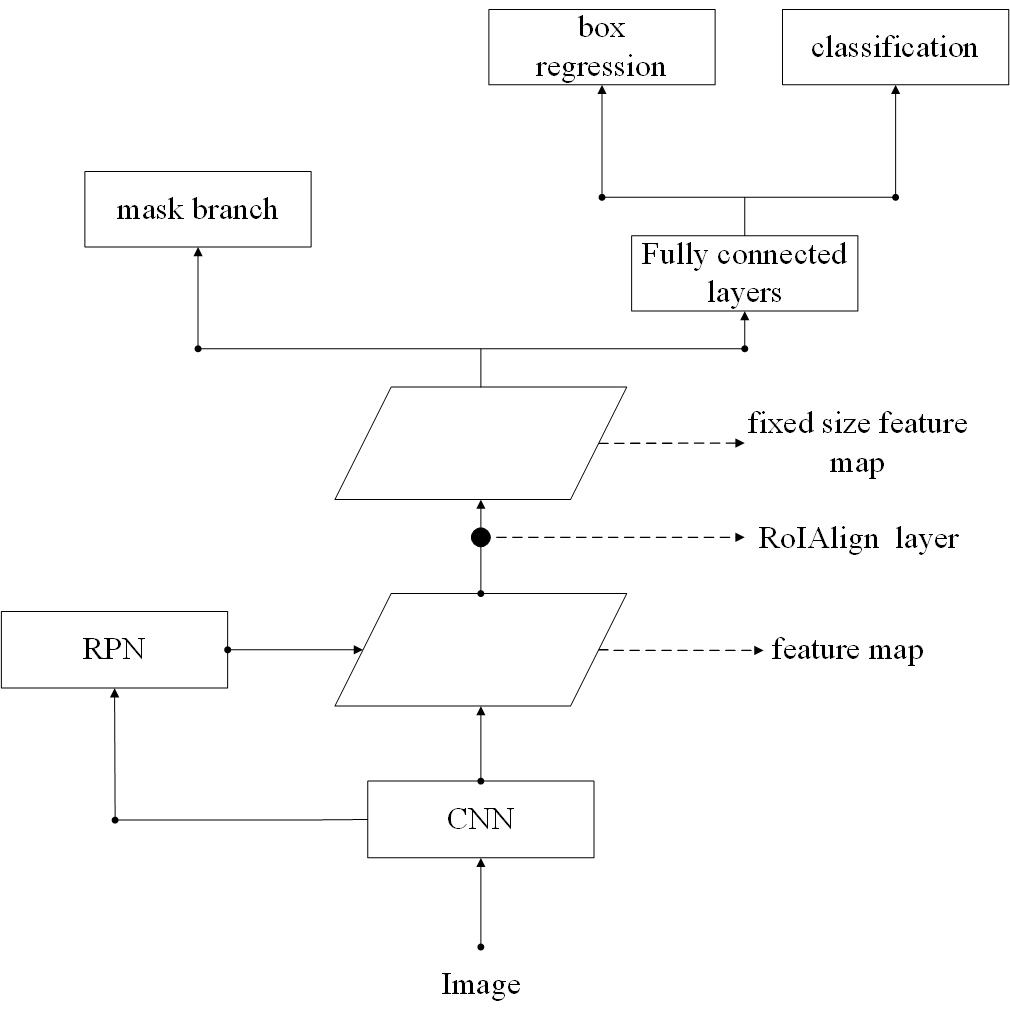


图 3.1 Mask R-CNN整体架构

backbone是一系列的卷积层，作用是提取图像的feature maps。深度学习中神经网络越深,学习的效果会更好，但构建太深层的网络后又会使得网络退化。ResNet（深度残差网络）则能在保持较高模型准确率的同时训练更深层的网络

本文使用的Mask R-CNN算法选取ResNet101为backbone，它的整体架构如图3.2所示。

ResNet分为5个stage，每个Stage的输出分别为C1-C5，这些输出将作为后面的FPN的输入。

在ResNet101中只存在两种类型的block：indentity block，conv block。

indentity block的结构如图3.3所示，输入x直接连接到第三个卷积层的输出，与x经过三个卷积层的输入相加后再通过激活ReLu。

conv block的结构如图3.4所示，它与indentity block的结构相差无几，只在shortcut

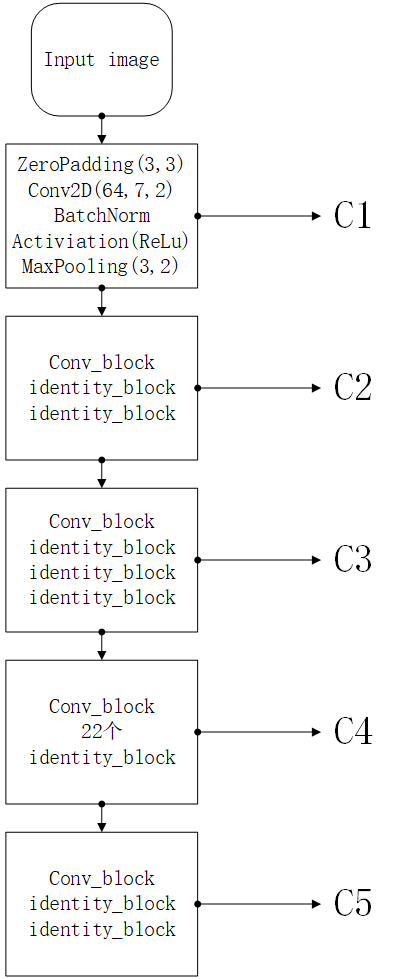


图 3.2 Resnet101结构图

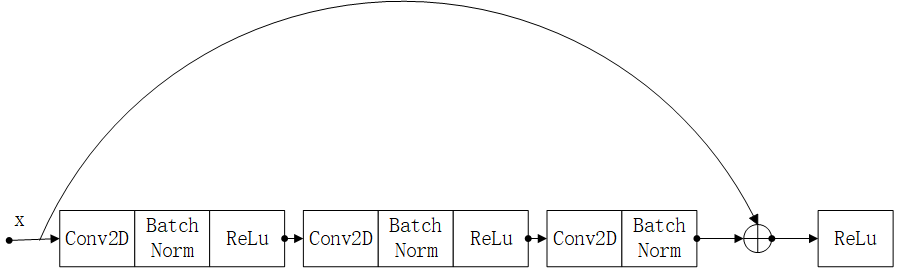


图 3.3 indentity block结构图

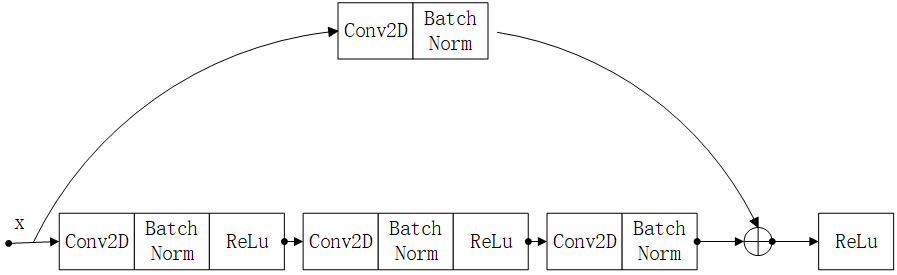


图 3.4 conv block结构图

上增加了一个卷积层。

FPN（Feature Pyramid Network）可以更好的将各层的feature maps利用起来，一般的网络或者较早的多尺度金字塔网络都是直接使用最后一层的feature maps，虽然能获得较强的语义，但位置和分辨率都比较低，对检测小物体比较困难。FPN融合了从底层到高层的feature maps，充分利用了ResNet101提取到的各阶段特征（C2-C5）。

FPN的结构涉及从下而上的路径，从上而下的路径，还有横向连接，结构如图3.5。

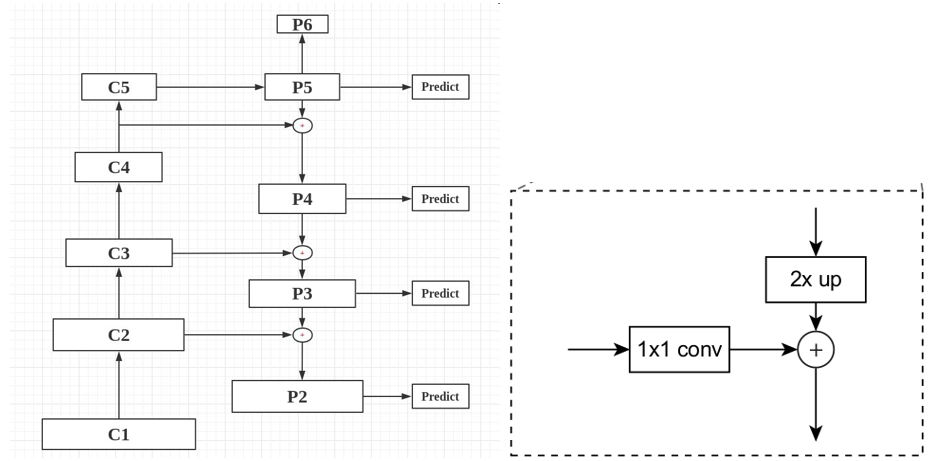


图 3.5 FPN结构图

图中的加号并不代表单纯的相加运算，而是左边的底层特征层通过11的卷积得到与上一层特征层相同的通道数，上层的特征层通过上采样（最邻近插值法）得到与下一层特征层一样的长和宽再进行相加，从而得到了一个融合好的新的特征层。

得到的P2-P5用于预测物体的bbox，box regression，mask，而P2-P6用于训练RPN。

RPN（Region Proposal Network）用于生成候选区域（proposals），RPN的头部结构如图3.6。

conv feature map为FPN得到的 P2-P6，anchors由三个参数确定：框的大小anchor\_scale，框的宽高比anchor\_ratio，框的中心像素点每次移动的距离anchor\_stride。对于每个sliding window的像素点，根据三个参数生成*k* 个anchor boxes，如P6大小为32×3216×16，保持面积不变，宽高比为(0.5,1,2)，然后得到3个anchor。

在算法中，使用单独的anchor\_generator产生anchors，P2-P6分别对应不同的anchor\_scale,得出所有特征图的anchor，返回值为张量（anchor\_count，()）。

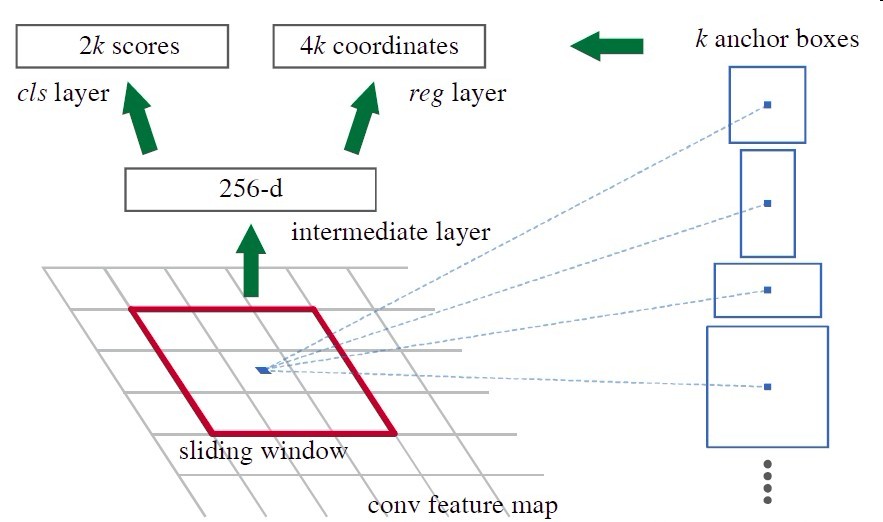


图 3.6 RPN头部结构图

intermediate layer代码中使用的是512d的conv中间层，对于*k* 个anchor boxes，通过1×11×1的卷积获得2*k* scores和4*k* coordinates。

*k* 个anchors通过softmax分类得到它的前景置信度和后景置信度，从而有2*k* scores。

4*k* coordinates包括每个anchor对应的四个坐标偏移量。由于正样本的位置与ground\_truth的位置不是完全一致的，可能存在偏差，所以在训练RPN之前需要修正anchor的位置和大小。这里要说到边框回归（bounding box regression）。

在anchors中的正样本中包括了目标物体，后续的分类即使正确地被识别出来了，但由于定位不准，也相当于没有正确的检测出目标物体。这个时候需要对box进行微调，使它和gt\_bbox更接近。

边框通常使用四维向量（x,y,w,h）表示，四个参数分别表示边框的中心点和宽高。要进行微调，需要找到一个映射，使，并且

。其中P为需要微调的边框，G为对应的gt\_bbox。

从边框P变换到gt\_bbox可以通过平移和尺度缩放实现，这也是R-CNN提出的，平移如公式3.1，3.2所示，尺度缩放如公式3.3，3.4所示。

(3.1)

(3.2)

(3.3)

(3.4)

此时只需要学习四个变换，，，。当边框P和gt\_bbox的IoU 0.6时，可以认为两者之间是线性变换，通过线性回归对边框微调。

线性回归就是给定输入特征向量，学习一组参数，使经过线性回归的值与真值无限接近。在边框回归中，输入的是边框对应的CNN特征，真值是边框P和gt\_bbox的平移量和尺度缩放构成的四维向量，真值的计算如公式 3.5~3.8所示。

(3.5)

(3.6)

(3.7)

(3.8)

损失函数如公式3.9所示，其中为要学习的参数，为输入的特征向量，为真值。

(3.9)

在训练阶段，根据输入的和得出，在预测阶段，根据输入的和训练得出的计算出从而实现边框微调。

RPN返回两个参数rpn\_probs，rpn\_bbox。rpn\_probs包含每个anchor的前后景置信度，rpn\_bbox为预测anchor的偏移量，这两个参数和anchors\_generator生成的anchors输入到Proposal layer用于生成更好的anchor。

Proposal layer首先根据前景score选取最多前6000个anchor，然后使用对应的偏移量对anchor进行位置修正并滤除其中越界的anchor，最后对anchor进行非极大值抑制（NMS），筛选后的anchor及对应坐标放入rpn\_rois作为返回值。

DetectionTargetLayer将rpn\_rois和从数据集读取的gt\_boxes，class\_ids，masks作为输入，首先剔除rois中同时包括几个实例的box（crowd box）不参与训练，再通过计算rois与gt\_boxes的交并比（IoU）选出数量比为1：3的正负样本用于训练，设置与一个gt\_box的IoU大于0.5的roi为正样本，与每一个gt\_box的IoU都小于0.5的roi为负样本。最后为每一个roi寻找一个的对应mask，返回对应roi的实际类别，对应roi的实际边框偏移量和对应mask。IoU的计算如公式3.10。

（3.10）

RoIAlign的工作原理如图3.7。

RoIAlign的输入由两部分，一个是RPN生成的rois，另一个是Resnet101产生的fearure map。conv layer使用VGG16，feat\_stride=32(图片经过VGG16缩小为原图的1/32)。若原图一个region proposal为m\*m，映射到特征图中的大小为（m/32）\*（m/32），不需要取整，保留浮点数。假定pooled\_w=n,pooled\_h=n，即是pooling后变为n\*n的特征图，所以region proposal映射到特征图大小为（m/32）\*（m/32）的feature map被划分为n\*n个小区域，每个小区域的大小为(m/(32\*n))\* (m/(32\*n)。若设置采样点为4，则将每个小区域平分为四份，每份取中心



图 3.7 RoIAlign原理图

点位置，中心点位置的像素值采用双线性插值法计算，这样一个小区域将得到四个点的像素值。最后取每个小区域四个点像素值的最大值作为这个区域的代表，最后得到nn的feature map，示意图如图3.8。



图 3.8抽样区域与双线性插值

RoIAlign将rois在对应feature\_map特征层进行特征提取，提取的特征图输入classifier。

classifier利用之前检测到的RoI进行分类和回归，输出物体检测的classes和bounding boxes，它的结构如图3.8所示。

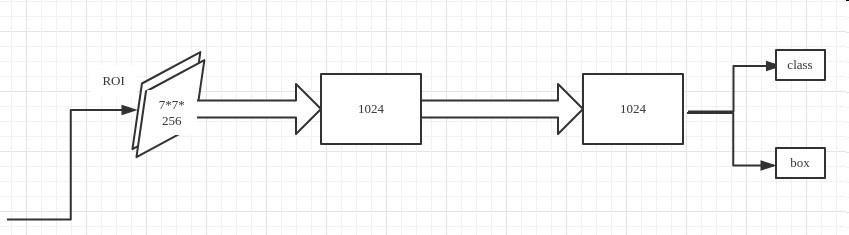


图 3.9 classifier的结构

RoIAlign以不同输出维度提取的rois的特征用于预测mask，mask的预测通过FCN（Fully Convolution Network）进行。训练的时候，classifier和mask同时进行；预测的时候，先得到classifier的结果，再将结果传入到mask预测分支得到mask。

Mask R-CNN的实现使用Github上Matterport的代码。

# 模型训练

## 4.1环境配置

算法使用python语言实现，首先需要配置编写环境。依次安装python包管理器anaconda3，python IDE pycharm3,安装完毕后开始配置本地python环境。

首先Create New Project,在anaconda目录下创建文件夹作为Project的存放位置，将python interpreter设置为已安装的python解释器，如图4.1。

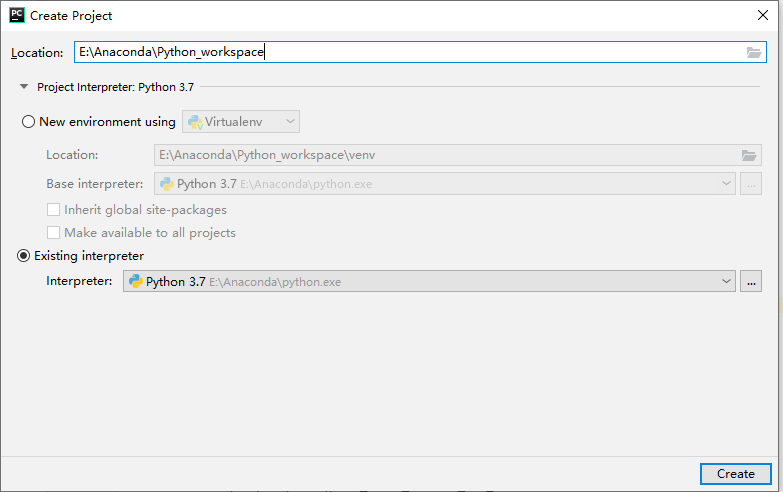


图 4.1 python解释器配置

第二步进行各种python包的安装，在pycharm主页面中点击左上角的File->Settings->Project->Project Interpreter打开pycharm的包管理器，需要安装的包如表4.1。

表 4.1需要安装的python包

|  |  |
| --- | --- |
| Packages | Version |
| numpy |  |
| scipy |  |
| Pillow |  |
| cython |  |
| matplotlib |  |
| scikit-image |  |
| tensorflow | >=1.3.0 |
| keras | >=2.0.8 |
| opencv-python |  |
| h5py |  |
| imgaug |  |
| IPython[all] |  |
| Pyqt5 |  |
| Pyqt5-tools |  |
| yaml |  |

所有的python包安装完毕后，将从github下载的Mask\_RCNN文件夹放入工程目录下，到此代码运行的环境已经准备完毕。

## 4.2数据集制作

环境配置完毕后，需要进行数据集的制作以供算法训练使用。本文使用标注工具labelme，labelme是麻省理工的计算机科学和人工智能实验室研发的图像标注工具，可以使用它执行图像标注。

首先安装labelme，运行Anaconda Prompt并依次执行命令conda create –name=labelme python=3.7，activate labelme,pip3.7 install labelme，等待一段时间后labelme就可安装完毕。

为了方便后续文件的命名及读取，还有规范输入图片的尺寸，需要对图片进行批量重命名和resize，主要代码如下：

oldname = os.path.join(images\_dir, j)

newname = os.path.join(images\_dir, 'rgb\_' + str(n) + '.jpg')

os.rename(oldname, newname)

im = Image.open(images\_dir + 'rgb\_' + str(i) + '.jpg')

im\_size = im.size

if (im\_size[0] > im\_size[1]):

im = im.resize((1280, 896))

第二步开始图像的标记，本文在中国腐蚀与防护网（China Gateway to Corrosion and Protection）上选取了178张金属腐蚀图片作为训练集。打开Anaconda Prompt或者命令提示符，输入labelme,按下回车键即可启动labelme，界面如图4.2所示。

labelme的使用非常简单，open dir设置打开文件的路径，file->chang output dir设置输出文件的路径（默认路径为标记文件的所处路径），点击Edit plolygons后可以左键选中已标记的label进行平移或者删除操作，右侧的label list为已存在的标签列表，polygon labels为本张图像标记的label。读取图片后右键可以选择创建label的形状，例如圆，矩形，多边形等，当连线为一个闭合连通区域后即可给label命名，本文训练的金属腐蚀缺陷有四种：pitting（点腐蚀），filiform(丝状腐蚀),uniform(一般攻击腐蚀)，crevice(缝隙腐蚀)。

标记完成一张图片后点击save会生成一个json文件，为了得到算法能够读取的数据，还需要在命令行执行命令labelme\_json\_to\_dataset dir,dir为单个json文件的路径，执行完成后会在json文件相同目录下生成一个文件夹，里面包含了若干文件，如图4.3，其中的yaml文件和位深为8的label.png文件是下一阶段训练模型所需要的数据，它们包含了图片中label的类别信息和位置信息。

由于每次只能转换一个json文件，效率较低，编写一个批处理py文件，主要代码如下：

path = 'F:/python/ErosionData/test'

json\_file = os.listdir(path)

for file in json\_file:

cmd = str('labelme\_json\_to\_dataset' + ' ' + path + '/' + file)

os.system(cmd)

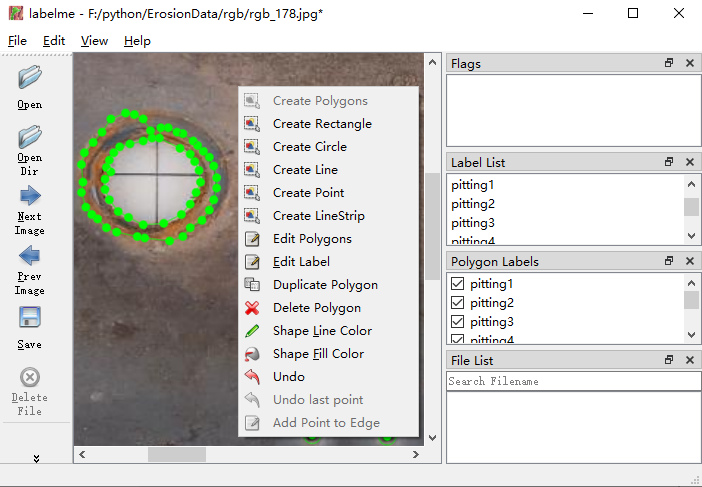


图 4.2 labelme标记物体

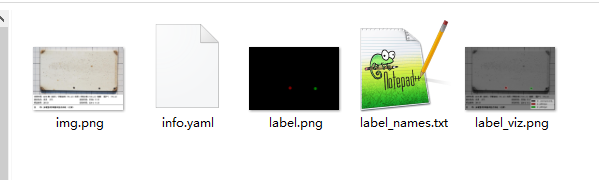


图 4.3 json文件夹

同样的，为方便后续读取文件，将所有生成的位深8位的png文件放入同一个文件夹中，批处理代码如下：

count = os.listdir(files\_dir)

for i in range(1, len(count) + 1):

im = Image.open(files\_dir + 'rgb\_' + str(i) + '\_json/' + 'label.png') # open files named like 'i'

im.save(target\_dir + 'rgb\_' + str(i) + '\_json\_label.png')

至此，数据集的制作完成。

## 4.3模型训练

模型训练的代码在samples文件夹下的train\_shapes.ipynb的基础上加以修改，由于ipynb文件无法在pycharm上运行，使用jupyter-notebook将train\_shapes.ipynb转换为py文件，操作如图4.4。

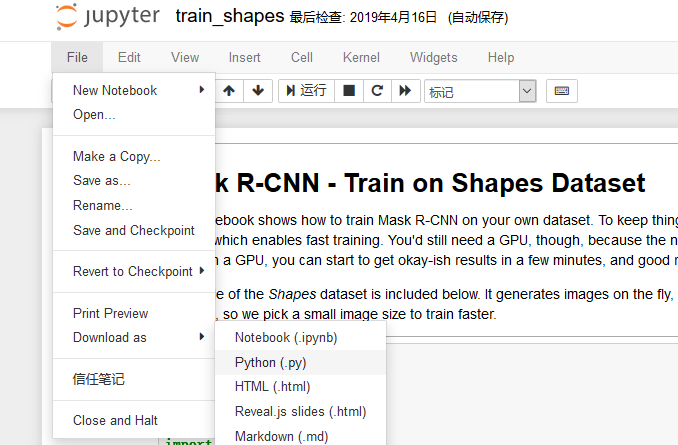


图 4.4 ipynb文件转换为py文件

在制作数据集中，算法已经将图片resize成了1280896大小，要训练的金属腐蚀缺陷有4类，加上背景总共是5类，主要新建一个训练类和修改基础设置，参数设置如表4.2所示。

表 4.2训练参数设置

|  |  |
| --- | --- |
| 参数 | 值 |
| BACKBONE\_STRIDES | [4, 8, 16, 32, 64] |
| DETECTION\_MAX\_INSTANCES | 100 |
| DETECTION\_MIN\_CONFIDENCE | 0.7 |
| FPN\_CLASSIF\_FC\_LAYERS\_SIZE | 1024 |
| IMAGE\_CHANNEL\_COUNT | 3 |
| IMAGE\_MAX\_DIM | 1280 |
| IMAGE\_MIN\_DIM | 896 |
| LEARNING\_RATE | 0.001 |
| NUM\_CLASSES | 7 |
| ROI\_POSITIVE\_RATIO | 0.33 |
| RPN\_ANCHOR\_RATIOS | [0.5, 1, 2] |
| RPN\_ANCHOR\_SCALES | (8, 16, 32, 64, 128) |
| RPN\_ANCHOR\_STRIDE | 1 |
| RPN\_NMS\_THRESHOLD | 0.7 |
| RPN\_TRAIN\_ANCHORS\_PER\_IMAGE | 256 |
| TRAIN\_ROIS\_PER\_IMAGE | 32 |

然后运行train\_shapes.py，等待片刻可以看到python控制台显示出训练配置，紧接着弹出一个标记过的图片读取示例，如图4.4，它将一张图片上的四个类别的label分开显示在四张图中，确认无误后关闭图片窗口。

对图片集的训练进行了两次：第一次设置的两个参数STEPS\_PER\_EPOCH =100，VALIDATION\_STEPS = 5，训练时长为4个小时左右；第二次保持其它参数不变将

上述两个参数改为了STEPS\_PER\_EPOCH=1000，VALIDATION\_STEPS=50，训练时长为23个小时左右。对应生成了两个模型：模型1和模型2。

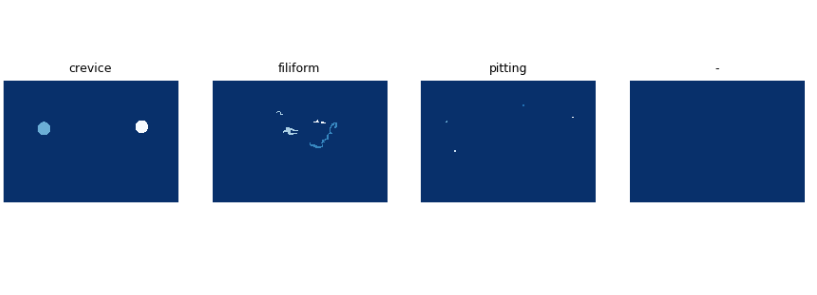


图 4.5数据集读取实例

模型生成后进行图片测试部分的准备工作，主要功能是调用模型生成结果并保存，同时将需要测试的图片路径设置为传入参数，参数由用户由UI界面传入，测试主要代码如下：

config = InferenceConfig()

model = modellib.MaskRCNN(mode="inference", model\_dir=MODEL\_DIR, config=config)

model.load\_weights(COCO\_MODEL\_PATH, by\_name=True)

class\_names = ['BG', 'pitting', 'crevice', 'filiform', 'uniform']

image = skimage.io.imread(img\_dir)

results = model.detect([image], verbose=1)

r = results[0]

visualize.display\_instances(image, r['rois'], r['masks'], r['class\_ids'],

class\_names, r['scores'])

plt.savefig('test.png', bbox\_inches="tight")

# UI界面设计

## 5.1GUI工具选择

UI界面的设计可以使用多种语言完成，例如C++,java等等，由于算法部分的代码是用python语言完成的，为了便于代码的管理，减少算法部分与UI部分的不必要连接，UI界面的设计也使用python语言完成。

在python语言诞生之后，许多优秀的GUI工具整合到了python当中，这些GUI工具集使得python也可以在图形界面编程领域占据一席之地。由于python非常流行，许多应用程序都是由python语言结合一些优秀的GUI工具集编写而成的。常见的python GUI编程工具集有五个：Tkinter，wxPython，PyGTK，PyQt，PySide。

Tkinter是python的Tk GUI工具包的接口，它可以在大多数Unix平台下运行，也可以在Windows和Macintosh系统上使用。它实际上是python包装的Tcl代码，调用Tkinter的操作会转换成命令，然后通过Tcl解释器进行解释，从而完成python的GUI界面。它的优点是历史悠久，是python真正标准的GUI，对多系统多平台的兼容性，而且上手快，用起来简单。

wxPython是以跨平台GUI库wxWidgets的python封装的形式提供给用户的，和python一样也是一款开源软件，拥有非常的跨平台能力，能够在绝大部分Unix系统，32/64位Windows，Macintosh OS下正常运行。wxWidgets使用标准的C++编写而成，可以与现有的各类工具库无缝连接。但由于wxWidgets缺乏较好的商业化支持，开发文档和资源等都不充足。

PyGTK也拥有跨平台性，它可以直接地运行在各种操作系统上而不发生异常，例如Linux，Windows，MacOS。除了简单易用和快速的原型开发能力外，PyGTK还有一流的处理本地化语言的独特功能。除此之外，PyGTK是自由软件，一种不受限制地自由使用，修改，研究的软件。

Pyqt是一个可以创建GUI应用程序的包，它使用python语言编写而成，融合了Qt库。Qt是用不标准的C++实现的，与其他库的兼容性会有问题，在每个平台的图形界面也并不完全是原生界面（Native GUI），只是透过theme去模拟系统上的标准GUI，所以看起来很像，有些地方则会明显看出破绽，Qt的执行速度缓慢是另一个问题。但Qt库是所有GUI工具库中最面向对象的，同时也最稳定。

Pyside是由诺基亚公司开发的，简单来说是Qt库的python移植。PyQt比Pyside出现得早，但具体应用两者都差不多，实现相同的功能PyQt和Pyside的操作都相差无几。PyQt的个人应用免费但商业应用收费，而Pyside则实行个人和商业的双免费策略。

将五个GUI工具集的优缺点互相比较可以发现，Tkinter的优点在于完成的GUI最标准，wxPython和PyGTK都有优秀的跨平台能力但wxPython的资源匮乏，PyQt最面向对象且最稳定，Pyside最大优点是个人与商业双免费。

由于本文的GUI开发不需要与商业无关，同时也不需要跨平台运行，在GUI的标准性方面也没有要求。但面向对象性十分重要，它将对象作为程序的基本单元，把程序和数据封装在里面从而提高软件的重用性，灵活性，扩展性，除此之外，GUI工具的稳定性也很重要。

综合以上因素，最终选择PyQt作为本文的UI开发工具。

## 5.2环境配置

PyQt包含许多模块，其中QtCore模块用于文件，目录，时间，线程和进程，QtGui模块包括图形相关的类，用于窗口，按钮，字体，颜色等等，在本章的UI界面设计中主要使用QtCore和QtGui这两个模块。

Pyqt5和Pyqt5-tools的安装已经在模型训练的环境配置中完成了，接下来还需要在pycharm里进行配置。打开文件->设置->工具->外部工具，点击加号进入create Tool界面。

首先添加Qt Designer，在Name栏输入Qt Designer，Program栏输入designer.exe的路径，Arguments栏输入$FileDir$\$FileName$，Working diretory栏输入$FileDir$。

然后添加pyuic5,其它栏的输入与之前的相同，Arguments栏改为$FileName$ -o $FileNameWithoutExtension$.py。

最后添加pyrcc，其它栏的输入与之前的相同，Arguments栏改为$FileNameWithoutExtension$.py。

至此Pyqt的环境配置完成，点击pycharm主菜单的工具->外部工具->QtDesigner进入UI设计界面。

## 5.3窗口设计

Qt Designer是用于使用QtWidgets设计和构建图形用户界面（GUI）的Qt工具，可以通过直接拖拽各种组件进入设计的UI界面，相比较而言，通过代码生成组件和设置组件的大小，长宽等属性十分复杂，而且每次都需要运行py文件显示出UI界面才能根据结果调整位置。Qt Designer则是反过来根据绘制好的UI界面生成代码，省去了调整每个组件的位置和整体布局问题。

Qt Designer一共有四种布局：垂直布局，水平布局，格子布局，表单布局。垂直布

局内的控件按垂直方向分布，水平布局内的控件按水平方向分布，格子布局中行和列交叉形成若干个格子，每个格子内可以放置一个控件，表单布局中的控件以两列的形式布局在表单中，左列用来放置标签，用于提示信息，右列相应的放置输入控件，如QLineEdit等。

首先制定UI界面拥有的功能：

（1）使用说明：给用户使用提示

（2）最小化：可以让界面最小化

（3）最大化：可以使界面最大化

（4）退出：可以退出界面

（5）浏览按钮：选择需要进行检测的金属腐蚀图像

（6）测试按钮：UI界面的核心功能，对选择的金属腐蚀图像进行检测

（7）进度条：在后台进行金属腐蚀图像的检测时，优化用户的等待体验

各项功能制定完毕后开始控件的布局：首先创建一个垂直布局，放入一个label控件存放金属腐蚀检测的说明。紧接着在垂直布局下方放置一个水平布局，放入两个控件，一个lineEdit控件用来显示浏览按钮选择图片的路径，一个label控件对lineEdit控件做补充说明。然后再放置一个水平布局放入进度条控件，最后把所有按钮放入一个垂直布局中。总的布局如图5.1所示。

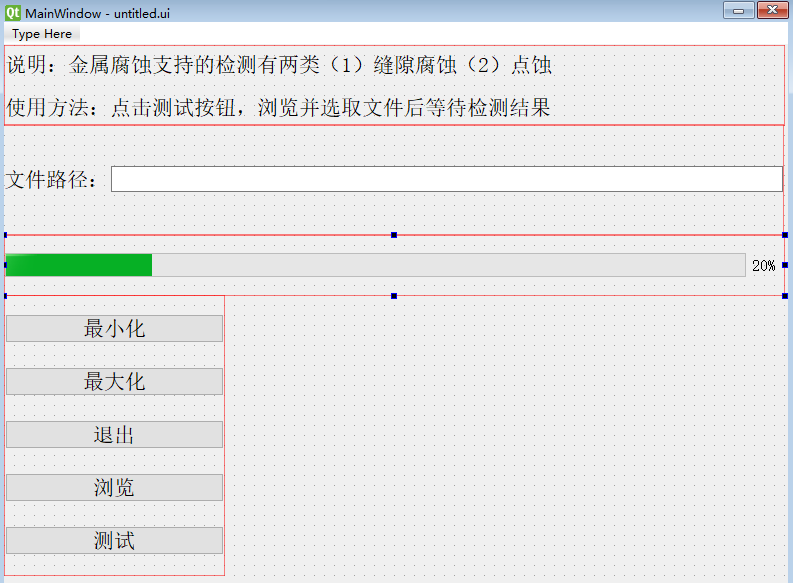


图 5.1控件的布局

保存untitled.ui文件后，在pycharm中右键untitled.ui文件使用外部工具pyuic5生成ui文件对应的py文件。

生成的py文件不能运行，还需要将类实例化和初始化，main.py代码如下：

class Mywindow(QtWidgets.QMainWindow, untitled.Ui\_MainWindow):

def \_\_init\_\_(self, parent=None):

# super(Mywindow, self).\_\_init\_\_()

QtWidgets.QMainWindow.\_\_init\_\_(self, parent)

self.setupUi(self)

app = QtWidgets.QApplication(sys.argv)

window = Mywindow()

window.show()

sys.exit(app.exec\_())

实例化之后可以显示UI界面，然后开始为按钮添加槽函数，在这之前需要说明pyqt中信号与槽的机制。信号与槽是Qt中的核心机制，同样也是PyQt中各个对象之间通信的机制。数据传输一般的流程为需要执行操作的类发出信号，另一个与信号连接的的类接收到信号，接收方调用槽函数执行操作。一个信号可以连接多个槽，同样的一个槽也可以监听多个信号。

首先给最大化，最小化，退出三个按钮添加槽函数，用Qt Designer打开untitled.ui进入编辑信号与槽界面，分别把showMaximized（），showMinimized（），close（）三个系统函数连接至三个按钮。将浏览按钮的槽函数设为view（），测试按钮的槽函数设为test（）。

第一步写浏览按钮的view()函数，由于测试集在工程目录下，设置浏览打开的起始界面为工程根目录。为防止运行错误，限制选取的文件类型只有三种：jpg文件，png文件，bmp文件。在Mywindow类下定义函数，代码如下：

def view(self):

file, filetype = QtWidgets.QFileDialog.getOpenFileName\

(self, "open file", "E:/Anaconda/Python\_workspace", "JPG Files (\*.jpg);;PNG Files (\*.png);;BMP Files (\*.bmp)")

self.lineEdit.setText(str(file))

接下来写测试按钮的test（）函数，函数需要完成调用腐蚀模型预测图片并生成结果返回给UI，由于图片较大，这里新建一个result.ui文件用于显示结果，整个ui文件内只有一个label控件用于存放图片。但直接按上述流程完成test（）函数会出现两个问题：（1）模型预测的工作量较大，时间较长（接近30s），会造成UI主界面显示未响应（实际上后台仍在运行，等待一段时间后会显示结果）（2）进度条需要在开始调用模型的时候开始计数，但模型调用和进度条处理无法在一个函数里并行执行。解决这两个问题需要用到多线程。

在大型软件的开发中，多线程是一种常见的编程技术。使用多线程可以将主线程（UI线程）中耗时较久的操作移动至单独的线程处理，避免这些操作阻塞主线程造成软件界面假死现象，更严重时假死的UI界面会被窗口处理程序检测到而被强制关闭。

现在完成测试和进度条功能的解决方法就十分明显了，首先在test（）函数里创建一个子线程，线程内读取浏览按钮得到的图片路径并传入测试函数。

与此同时，test（）函数启动进度条，子线程内图片预测完毕生成结果后，发射信号，与信号相连的callback（）函数接受到信号后立即将进度条充满，然后将生成的结果在子窗口result.ui上显示。主要实现的代码如下：

self.timer = QBasicTimer() # 初始化一个时钟

self.step = 0 # 进度条的值

self.pbar.setValue(0)

def test(self):

self.timer.start(200, self) # 启动QBasicTimer，每200ms调用一次事件回调函数

self.thread = Example()

self.thread.start()

self.thread.signal.connect(self.callback)

def callback(self):

self.pbar.setValue(100)

pix =QPixmap('E:/Anaconda/Python\_workspace/Mask\_RCNN/UI/test

.png')

child.show()

child.label.setPixmap(pix)

child.label.setScaledContents(True)

def timerEvent(self, a0: 'QTimerEvent'):

# 把进度条每次充值的值赋给进度条

self.pbar.setValue(self.step)

if self.step == 99:

# 停止进度条

self.timer.stop()

self.step = 0

return

# 把进度条卡在99，等处理好了再到100

if self.step < 99:

self.step += 1

class Example(QThread):

signal = pyqtSignal()

def run(self):

path = window.lineEdit.text()

detection(str(path))

self.signal.emit()

UI界面各个控件的功能到此已全部实现，各个控件之间的联系和整体结构如图5.2。

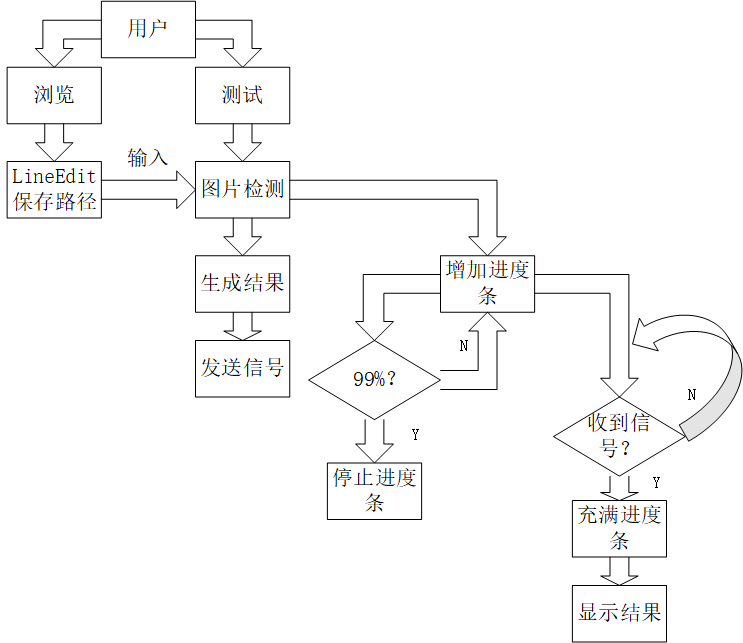


图 5.2 UI控件的结构图

# 系统测试

## 6.1系统功能测试

运行main.py,出现UI主界面，分别点击最大化，最小化，退出按钮可以看到界面最大化，最小化，退出，这三个按钮的功能正常。

点击浏览按钮，可以看到弹出的文件选择窗口，可选择的文件格式如设定的三种图片格式，选择图片后点击确定，lineEdit控件正常显示图片路径，如图6.1所示。

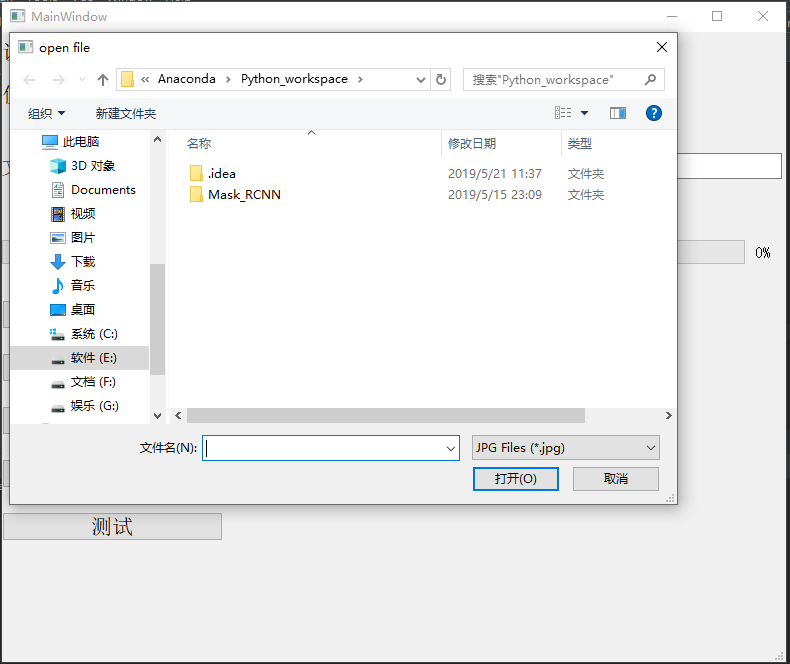


图 6.1浏览按钮测试

点击测试按钮后，进度条开始启动，等待进度条达到100%时，弹出了一个新的窗口显示图片，进度条如图6.2所示，结果如图6.4所示。

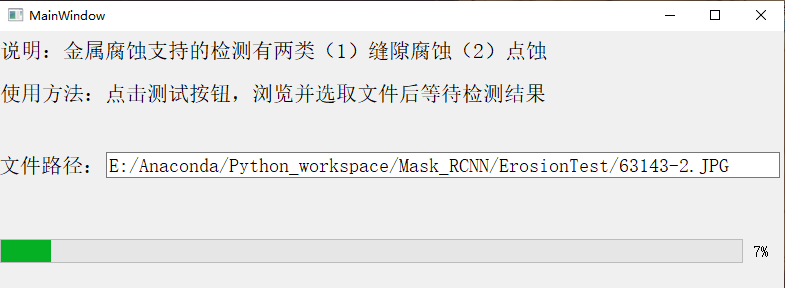


图 6.2进度条测试

在前面的模型训练部分中，生成了两个模型：模型1和模型2。用同一张图片对两个模型分别进行测试，输出结果分别为图6.3和图6.4。

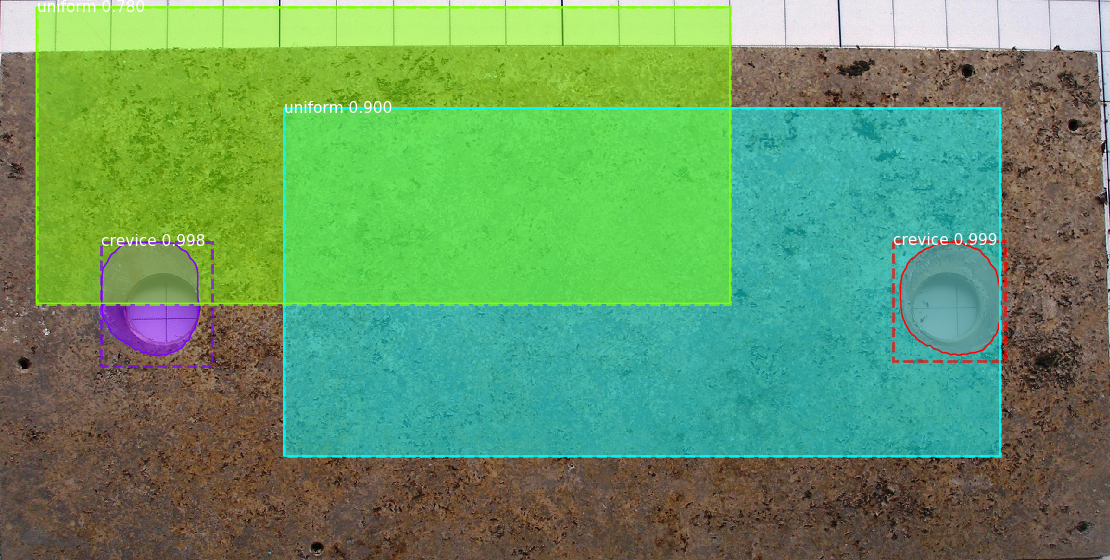


图 6.3模型1测试结果



图 6.4模型2测试结果

通过测试可以看到：模型1和模型2对一般攻击腐蚀预测的位置都不准确，但明显模型1的位置偏差更大；模型1对这张图片中的四个腐蚀孔完全没有检测到，模型2则全部很好的标记了出来。可以得出训练每轮迭代次数为1000，验证集每轮迭代次数为50模型的识别效果比两个参数为100和5要好的多。总的来说腐蚀模型对点蚀和缝隙腐蚀的识别效果较好，但识别不出丝状腐蚀，而对一般攻击腐蚀则会产生误识别。

## 6.2系统指标测量

首先是测试检测一张图片所花费的时间。点击测试按钮，记录时间栏内的数据，如此反复，计算出平均值作为一张图片花费的时间。以25601792大小各不相同的八张图片测试了8组数据，如表6.1。

表6.1 测试图片时间

|  |  |
| --- | --- |
| 图片 | 时间（s） |
| 1 | 33.30 |
| 2 | 31.36 |
| 3 | 30.26 |
| 4 | 27.91 |
| 5 | 33.05 |
| 6 | 32.26 |
| 7 | 32.69 |
| 8 | 32.49 |

其中图片4未被检测出任何腐蚀，属于特殊情况，将图片4的数据剔除后，七组数据求平均后得到平均时间为32.20s，超出预期目标2.2s。

# 总结

该课题主要使用Mask R-CNN算法，结合深度学习框架tensorflow完成一个检测与分类系统，对金属腐蚀图像中存在的金属腐蚀缺陷进行检测和分类。根据金属腐蚀的特点以及对腐蚀缺陷的可视化要求（能同时得到腐蚀缺陷的位置和标签），选择了能实现系统的算法Mask R-CNN，并设计UI界面，两部分共同组成金属腐蚀检测系统。

通过测试得到以下结论：本文设计的金属腐蚀缺陷分类系统可以较好的识别点蚀和缝隙腐蚀，对丝状腐蚀和一般攻击腐蚀的识别效果却不好，但不能说模型不适于识别这两种腐蚀。由于得到腐蚀数据本身较少，再加上许多用于训练的图片有杂质掩盖，丝状腐蚀的样本量和标记的标准性都远小于点蚀。一般攻击腐蚀的标记也受到金属表面杂质的影响无法正确标记，因而样本量也较少。虽然在测试中，模型能够很好的识别缝隙腐蚀，但由于训练和测试都使用同种类型的金属板，缝隙腐蚀只发生几个在固定位置，所以不能代表模型也可以识别发生在各种位置的缝隙腐蚀。

总的来说，本系统选择通用物体检测算法完成腐蚀缺陷检测的优点是：不需要手动调整太多参数，可以训练生成模型同时检测多种腐蚀缺陷；识别新的腐蚀缺陷类别只需要将对应样本送入模型训练即可，而不用根据每种腐蚀的特性对参数进行调整。缺点是：用于训练的样本数量要足够，在标记物体时也要尽可能的准确，样本不足和标记不准确都会造成模型预测精度下降；标记物体是个繁琐而且耗时的过程，在没有较好的硬件支撑下模型训练非常耗时。

限制系统预测的效果的最大因素还是样本数量和样本的好坏。有些腐蚀由于自身的定义，例如一般攻击腐蚀（在整个金属表面发生的均匀的腐蚀），在标记时难以确定边界，再加上腐蚀数据相比较常见物体数据较少，这就造成了有些腐蚀识别的很好而有些较差。除了上述因素，改变训练的迭代次数也会影响模型的好坏，迭代次数过少会使模型在训练集和测试集上的预测效果都比较差，迭代次数过多，会发生拟合，使得模型在训练集的预测效果非常好但在测试集的预测效果下降，因此找到一个合适的迭代次数也可以提高系统的性能。

# 参考文献

[1]郝芳.人工智能的分支及重点发展领域研究[J].现代商贸工业,2018,39(19):187-188.

[2]See Simon. Artificial Intelligence Computing for a Smart City [M]. Springer International Publishing, 2018.

[3]张玮. 金属腐蚀形貌特征提取用于腐蚀诊断的研究[D].大连理工大学,2004.

[4]罗晓琳.基于像素差异的金属腐蚀区域分割仿真研究[J].计算机仿真,2012,29(04):317-319+327. [5]李丙涛. 基于图像分析的材料表面腐蚀特征定量评价系统研究[D].重庆理工大学,2010.

[6]熊岩. 金属板材腐蚀区域自动提取技术的研究[D].解放军信息工程大学,2009.

[7]陶蕾. 典型金属材料和涂层体系自然环境腐蚀检测技术研究[D].天津大学,2009.

[8]贺东海. 基于图像识别技术的镁合金腐蚀与防护研究[D].重庆大学, 2009.

[9]纪钢,吴萍萍.在HSV下的材料腐蚀特征区域颜色与边界量化处理[J].重庆理工大学学报(自然科学),2016,30(01):63-69.

[10] 纪钢. 基于计算机视觉的材料外观腐蚀特征在线检测系统[A]. 西南财经大学信息技术应用研究所、《计算机科学》杂志社.2009国际信息技术与应用论坛论文集(下)[C].西南财经大学信息技术应用研究所、《计算机科学》杂志社:西南财经大学信息技术应用研究所,2009:4.

[11] 于淼. 基于计算机视觉的镀层腐蚀评级系统[D].沈阳工业大学,2007.

[12] Stankiewicz A, Winiarski J, Stankiewicz M, et al. Corrosion resistance evaluation of Ni‐P\nano‐ZrO2 composite coatings by electrochemical impedance spectroscopy and machine vision method[J]. Materials and Corrosion, 2015, 66(7): 643-648.

[13]Alicja Stankiewicz,Maciej Stankiewicz,Juliusz Winiarski,Irena Szczygiel,Bogdan Szczygiel. Machine Vision System for Corrosion Detection as an Additional Tool beside EIS for Evaluation of Protective Properties of Electrolessly Deposited Ni-P Coatings[J]. Solid State Phenomena,2015,3763(227): 557-560.

[14] Ramos R, Zlatev R, Valdez B, et al. LabVIEW 2010 computer vision platform based virtual instrument and its application for pitting corrosion study[J]. Journal of analytical methods in chemistry, 2013, 2013.

[15] Yammen S, Muneesawang P. An advanced vision system for the automatic inspection of corrosions on pole tips in hard disk drives[J]. IEEE Transactions on Components, Packaging and Manufacturing Technology, 2014, 4(9): 1523-1533.

[16] Petricca L, Moss T, Figueroa G, et al. Corrosion detection using AI: a comparison of standard computer vision techniques and deep learning model[J]. CCSEIT, AIAP, DMDB, MoWiN, CoSIT, CRIS, SIGL, ICBB, CNSA, 2016: 91-99.

[17] Zaidan B B, Zaidan A A, Alanazi H O, et al. Towards corrosion detection system[J]. International Journal of Computer Science Issues (IJCSI), 2010, 7(3): 46.

[18] Atha D J, Jahanshahi M R. Evaluation of deep learning approaches based on convolutional neural networks for corrosion detection[J]. Structural Health Monitoring, 2018, 17(5): 1110-1128.

[19] Zulkifli N A, Sahari K S M, Anuar A, et al. Corrosion detection using labVIEW for robotic inspection of boiler headers[C]//The 8th International Conference on Robotic, Vision, Signal Processing & Power Applications. Springer, Singapore, 2014: 31-37.

[20] Enikeev M, Enikeeva L, Maleeva M, et al. Machine learning in the problem of recognition of pitting corrosion on aluminum surfaces[J]. Information Technology and Nanotechnology,2018.

[21]He K, Gkioxari G, Dollár P, et al. Mask r-cnn[C]//Proceedings of the IEEE international conference on computer vision. 2017: 2961-2969.

# 致谢

本设计是在导师肖攸安教授的指导和支持下完成的。在整个毕业设计阶段，从文献调研到完成毕业设计论文，老师每周在会上在检查进度的同时，还会指出我们完成部分的不足并且给我们十分有意义的建议。每次在研究上遇到困难，老师都会孜孜不倦地给我们指导方向。能够最终完成这篇毕业设计，是少不了老师平日里耐心的解答和严格的督促的。在此谨向导师肖攸安教授致以衷心的感谢。

同时，在完成毕业设计的各个阶段，同班的同学也给予了我热心的帮助，使我在浓厚的学习氛围里受益匪浅，在此对他们表示感谢。

最后，要感谢我的父母和家人，从准备考研到完成毕业设计，他们对我关心至微，在我背后默默支持，是我能坚持下来强大的精神支柱。