队伍编号	MCB2300305
赛道	A

基于卷积神经网络模型对坑洼道路检测和识别的研究

摘 要

道路坑洼识别是一种利用计算机视觉识别数字图像的技术，本模型旨在利用卷积神经网络进行识别路面坑洼。问题 1 问要求建立合适模型对路面进行识别训练。问题 2 要求对模型进行多维评价，问题 3 要求使用训练后的模型对测试集进行识别。

问题 1 中，建立了一个相对简单的卷积神经网络模型对其进行训练。在数据输入前需要首先进行预处理。为了达到训练要求，需要打上坑洼与平坦的标签，接着进行高斯去噪处理以求减少图像噪点。通过增强对比度提高后期的处理精准度与可读性。进行灰度化压缩图层通道，增加模型精确度与计算速度并略微增加可读性。使用直方图均衡化以期平衡灰度分布，减少图像文件由于原本光照与曝光导致的整体灰度分布不均。最终锐化图像细节，统一降采样到 256×256 格式，完成预处理。最终进入输入层进行阈值化进行下一步卷积。设置了三层 3×3 大小的卷积核进行卷积，每层卷积核数量按 32，64，128 依次递增。大量但小分辨率的卷积核能提供更佳的精度。每一次卷积过后，会进行池化操作。第一次池化使用软池化操作，减缓图像特征扭曲，尽可能的保存图像关键信息。第二次和第三次池化采用最大池法，以在尽量高效率的情况下提取图像特征值。在除最后一步的过程激活函数，选用了 ReLU 整流线性单位函数，以期模仿生物神经元对消息编码稀疏分散的特性，达到更好的计算效率，减少计算量。最后输入到全连接层的神经元中，进行学习，输出一个分数进行判断，最终它是平坦还是坑洼图像。其中结果激活函数采用 Sigmoid 函数对此二分类模型进行评价，此函数能将数据投射到 0~1 中，方便分类图像。同时，考虑到卷积神经网络需要大量的样本进行训练，所提供的训练集数量不够充裕，因此引入了部分外部训练集进行训练。

问题 2 中，针对训练好的模型，从卷积神经网络本身的性质入手。采用准确率，精准率，召回率，F1 分数，ROC 曲线下面积（AUC），混淆矩阵以及交叉熵损失对模型进多维度评价。评价结果显示，准确率为 **0.8361**，精确率为 **0.6286**，召回率为 **0.6**，F1 分数为 **0.6921**，ROC 曲线下面积（AUC）为 **0.8490**，每 **0.3637** 单张照片的运行时间: **0.04115** 秒。在准确率、精确率和 F1 分数相对较高的情况下保持了相对较小的召回率，分类器在识别正样本和负样本时可能存在一定的误判和漏判，但总体来看尚在可接受范围内。ROC 曲线下面积（AUC）也接近于 1，没有表现出明显的随机性，计算速度也较快。总体来看，评价维度全面而客观，模型差强人意，可以使用。

问题 3 将测试集经过预处理后导入到训练好的模型中去进行判断，最后结果是平坦路面识别为 **683** 张，坑洼路面识别为 **4259** 张。

关键词：卷积神经网络 卷积 最大池化 软池化 全连接层

目录

1.问题提出

1.1. 问题背景

计算机视觉在路面道路坑洼检测中的应用是一个热门的研究领域。通过计算机视觉技术，可以利用图像或视频数据来自动检测和识别道路上的坑洼，从而提高道路维护的效率和安全性。

道路坑洼是由水和车辆行驶对路面的综合作用形成的结构性道路损坏。坑洼不仅给驾驶员带来不便，还对车辆状况和交通安全构成威胁。因此，定期检查和修复道路坑洼是必要且关键的。目前，人工目视检查仍然是道路坑洼检测的主要形式。结构工程师和认证检查员定期检查路面坑洼并报告位置。然而，这种方法效率低、昂贵且危险。

计算机视觉算法在路面坑洼检测中的应用已有二十多年的历史。近年来，随着深度学习等技术的发展，基于计算机视觉的路面坑洼检测取得了显著进展。但基于计算机视觉的路面坑洼检测仍面临一些挑战，比如复杂的道路环境、光照变化和噪声等。未来的发展趋势包括进一步改进基于深度学习的方法，如无监督学习多模态语义分割，以提高路面坑洼检测的准确性和鲁棒性。

1.2. 问题重述

·**问题 1**：根据题目所给图形，对图像进行灰度调整，噪点去除等预处理，进而提取图像特征，从而设法建立合适的模型，用来识别图像中的道路是坑洼还是平坦，并力求达到高准确率、高速、高稳定度的模型。

·**问题 2**：对问题 1 中的模型进行训练迭代，在不断精进模型的同时，从各个维度对模型进行全面的评价，审视其利弊。

·**问题 3**：利用训练好的模型，对所给测试集中的图片进行识别测试，辨别他们是坑洼道路还是平坦道路，并将结果保存，提交。

2.问题分析

2.1. 问题 1

将路面图像转换成数学表示，可以使用二维矩阵，其中每个元素表示一个像素点的灰度值。接着使用卷积核对原始图像进行卷积操作，提取图像的特征。卷积核可以看作是一个小型的滤波器，它在图像上滑动，对每个像素点进行卷积计算，生成一个新的特征图。并对卷积后的特征图进行池化操作，通常使用最大池化或者平均池化等操作，以减少特征图的大小，同时保留重要的特征信息。最后将池化后的特征图转换成一维向量，然后通过全连接层进行分类，确定图像属于哪个类别。

2.2. 问题 2

对模型进行了多次训练之后，分别计算准确率，精准率，召回率，F1 分数，ROC 曲线下面积（AUC），混淆矩阵以及交叉熵损失等参数计算它的模型数据。

3. 模型假设

·假设所有的图片均可分类到坑洼和平坦中去，不存在中间状态。
·训练集中不存在无法识别不属于“路面”的图像数据。
·文件中不存在无法阈值化的图像数据。

4. 符号说明

符号	符号说明
s_i	梯度直方图概率的累加值
$x_{i'j'k}$	输入特征图的第i'行、第j'列和第k个通道的像素值
y_{ijk}	输出特征图的第i行、第j列和第k个通道的像素值
$R_{i,j}$	输入特征图中第i行、第j列的区域
$x_{k,j}$	输入层第k个神经元的输出值
$w_{i,k}$	对应位置的神经元的权重
$y_{i,j}$	神经元在对应位置的输出
b_j	第j个位置的偏置值
$\tilde{a}$	特征值输出值
TN	分类器正确地将负例分类为负例的样本数
TP	分类器正确地将正例分类为正例的样本数
FP	分类器错误地将负例分类为正例的样本数
FN	分类器错误地将正例分类为负例的样本数

5. 问题 1：通过卷积神经网络建立坑洼路面识别的数学模型

5.1. 数据预处理

初步分析所给已标记图片，可以发现，这些图片大小不一；一些图片质量不佳，存在相当数量的噪点；一些拍摄焦距不适宜导致图片模糊；还有一些甚至有没去除的图片水印乃至广告，这些都会干扰后续模型对图像的处理，需要进行去噪。另外，这些图像在复杂度，对比度，以及一些模糊边缘的细节上并不利于本文所用卷积神经网

络模型的处理，因此，首先需要对所给图像数据进行预处理。

5.1.1. 标签分类

对提供的数据组进行分类，正常路面标签为 1，坑洼路面标签分为 0，置为训练集。

5.1.2. 去噪处理

正常在拍摄时，受制于现场环境、元器件状态，图片会产生噪点，这种噪点一般服从高斯分布，被称为“高斯噪声”。因此我们需要对图像进行高斯滤波处理。利用二维高斯函数进行消除操作：

$$G(x,y)=\frac{1}{2\pi\sigma^2}e^{-(x^2+y^2)/2\sigma^2}$$

这样，通过将像素点本身和领域内的其他像素点经过加权平均后得到的值置为像素点的值，从而尽量去除高斯噪声。

需要说明的是，为了避免除噪过度而导致喧宾夺主，这里不采用其他的除噪方式进行处理。另外，题目所给数据中有几张图片并非拍摄而来，而是类似于 CG 图的形式，这些图像不存在严重的高斯噪音，因此手动将它们挑出，以免除噪过度。

5.1.3. 增强对比度

对比度对生物视觉与计算机视觉的影响极为显著。直观来看，高对比度能够更好的凸显出图片的细枝末叶，这不仅增强了图像在视觉上的效果，也为模型识别提供了便利：更加丰富突出的图像细节可以允许识别模型设置一个较为妥当的置信度阈值，变相提高模型的准确性。尽管过高的对比度会导致色彩失真，但后续会对图像进行灰度化，因此在这一步操作中，可以适当较高的增加图像对比度。

图 1

图 normal305 原图与拉高对比度后的对比

5.1.4. 灰度化

一个正常图像中的色彩一般是由红黄蓝三原色组成，图片中每一个像素点都有其对应的 RGB 数据，但本模型所需读取的图像并不需要其颜色数据，同时，这些颜色数

据信息也需要额外的卷积核进行识别，从而导致模型识别速度下降，读取速度过慢。所以，出于减少内存占用，增加读取速度的考量，使用 cvtColor 函数对图像数据进行灰度化处理，设置其 R=G=B,以减少模型中卷积核的绝对数量，提高卷积核的使用效率。值得一提的是，灰度化处理也能略微提高图像对比度，增强模型效果。

5.1.5. 直方图均衡化

在对图像进行灰度化后，需要重新调整图像的灰度，以求提高整体对比度。在图像仅以灰度来表示信息时，由于原始曝光，亮度等问题，有必要重新分配其灰度分布的整体区间，改善图片观感与可读性，为模型提供更好的数据。

由 RGB 彩色图像转化而来的灰度图像，灰度级一般是 0~225。可以将原始的灰度值从 0~1 平均分为八个梯度（0 和 1 也算一个梯度），统计其像素数n_k与概率P_j，由公式：

$$s_k = T(r_k) = \sum_{j-0}^{k} P_r(r_j) = \sum_{j-0}^{k} \frac{n_j}{N}$$

可以依次算出各个梯度直方图概率的累加值s_i，将s_i与原灰度级分布近似的值相匹配归类，得到新灰度级分布，最终压缩分布级，得到一个相对均衡的亮度分布。便于模型识别处理。

5.1.6. 图像锐化增强与整体处理

在处理完上述步骤后，可以再对图像文件进行适当的锐化处理，以求尽可能更出色的可读性，出于减小对图片有效信息破坏的原则，只适当拉高锐化半径进行处理。

最后，为了保证程序的读取效率，我们将图片分辨率统一为 256×256 格式输入模型。这样，数据图像的预处理便完成了，最终处理完的图片，大大提高了数据的可读性，压缩了信息通路，提高了信息密度，加上图片的标签，大小不超过 10MB，这大大减少了模型的运算量与复杂程度，提高了运算速度。

图 2

图 normal305 原图与预处理后图的对比

5.2. 模型构筑

卷积神经网络（Convolutional Neural Networks, CNN），早在上世纪七八十年代就已出现，但受制于算法和硬件问题，并没能在与其他算法的竞争中脱颖而出，但在十年前作业，凭借准确率和计算效率的优势，卷积神经网络脱颖而出。卷积神经网络模型很擅长处理图像识别问题。

卷积神经网络是一种基于多层监督学习的人工神经网络，通过模拟人类对视觉信号的处理方式，尝试让计算机高效的处理接受到的信息。对于生物来说，视觉的识别原理是在观察到某种形状的图像时，大脑的固定区域便会产生脑电波，接着传递给上级脑区处理 。而卷积神经网络与此异曲同工，同样是由卷积层提取特征信号，使用激活函数加工，池化层加强特征，最终由全连接层计算分析，得到结果。

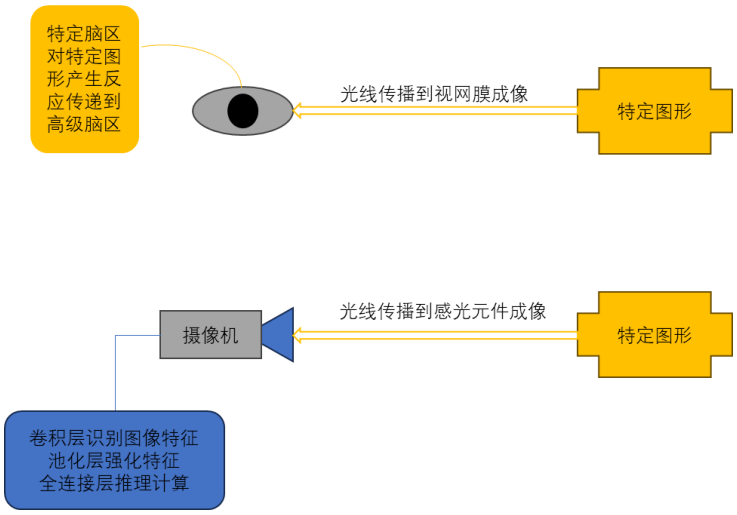

图 3
生物视觉神经网络与卷积神经网络对比

5.2.1. 输入层

在经过上文的预处理，特别是灰度化处理将 R，G，B 置为相等的值后，便能将图像传输给程序进行训练和识别，图像主要读取的是各个位置上的灰度值。如下图：

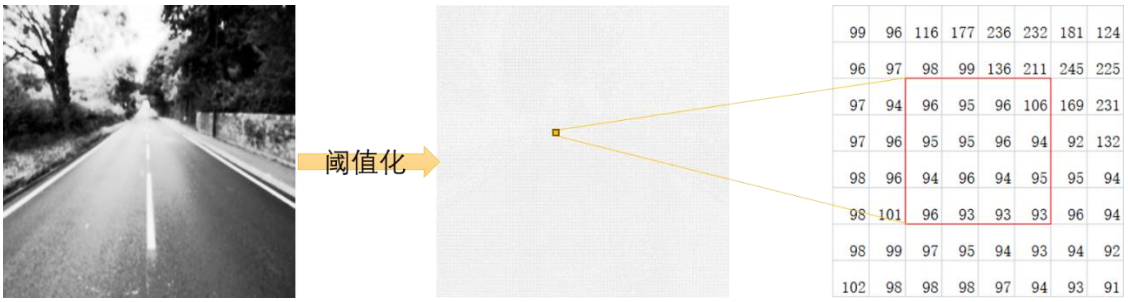

图 4
预处理后的图像与其转换来的灰度值矩阵

5.2.2. 卷积层

卷积层的作用是产生一组对应的平行特征图，通过在其上滑动特别设计的卷积核，执行计算来提取图像特征。在设计卷积层时，首先需要考量便是图像大小，以搭配合适数量的卷积核与卷积层。一般来说，更多的卷积核能得到更完备的特征信息，但也会拖慢计算速度，增加计算复杂度。但得益于近年来 GUP 性能的大幅提升，再加上我们在预处理时，已经将图片重新降采样到 256×256 大小，通路压缩到一层（R，G，B 三层通路均相等，视作一层通路计算），因而可以采取更加激进的策略进行卷积操作，以求得更高的准确度，设置更多层，每层更多数量的卷积核输出特征图。

与图像滤波算法的原理类似，卷积是指卷积核每次在特征图上点乘对应的子矩阵再加上偏置值（本模型中的偏置值默认为 0）从而得到一个特征图，这个特征图（卷积核）通常会小于等于图片分辨率，这也是卷积神经网络的特征之一。下一次卷积核会移动一定步长，依此方法再进行一次点乘运算，这样遍历整个图形，完成卷积化。

在第一次卷积时，每个卷积核设置为 3×3 大小，一共设置 32 个卷积核，在经过整流线性函数和池化处理后，输出形状减小为 254×254，再次进行卷积。第二次卷积核大小不变，但卷积核翻倍至 64 个，输出数据形状为 125×125，接着再进行整流、池化。第三次卷积同上，也是只将卷积核翻番至 128 个，输出数据形状为 60×60，其余不变。另需说明的是，在第一次卷积时，由于没有使用填充，每个边会丢失一个像素点。

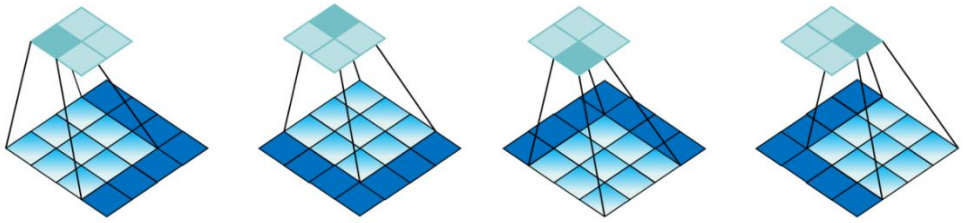

图 5
卷积过程原理图

5.2.3. 过程激活函数与结果激活函数

在每一步骤完成后，需要对特征图使用激励函数进行进一步处理。本模形选用了整流线性单位函数（Rectified Linear Unit, ReLU）：

$$f(x) = \max(0, x)$$

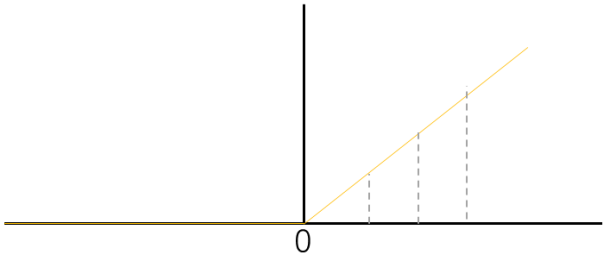

图 6
整流线性单位函数

ReLU 是非线性激活函数，这意味着它没有把输入映射到一个特定区间，而是进行

了一定的过滤处理。

这是为了模仿生物神经元对消息编码稀疏分散的特性，能达到更好的计算效率，没有使用正则化，极大的减少运算量。且在 x>0 时，不存在梯度消失问题，不会对模型的准确度造成太大影响，安全性较高。这样，得到处理妥当的特征图便能继续进入下一步的池化层。

在最终输出结果时，需要的是一个对于图像的度量值。我们取 Sigmoid 函数进行激活：

$$\sigma（x）= \frac{1}{1+e^{-x}}$$

它可以将任意实数映射到一个介于 0 和 1 之间的值。适合本例这样的二分类模型使用。此外，Sigmoid 函数是可导的，因此可以用于反向传播算法中，训练模型。

5.2.4. 池化层

池化本质上是一种非线性形式的降采样，它的作用是在尽可能保存特征图重要信息的同时降低图像的分辨率，便利计算。同时尽量凸出或者说展现特征图的重要信息。池化方法有最大池化，平均池化，Rol 池化等。

本题中需要突出坑洼道路的特征信息，因此要给池化层中最大值设置较高的权重。根据较新的文献资料，与实践表现，通常情况下最大池法的表现也更好[1]。但最大池法会导致图像存在一定程度的扭曲，特别是池化三次后扭曲是比较严重的，故此在第一次池化时使用软池化进行操作，软化池可以在简化过的激活图中保留相对较多的信息，更好的利用池化内核的激活因子，因而减少内存使用。可以提高算法精度。软池化的主要原理就是使用 softmax 进行加权池化，根据非线性特征值计算其矩阵权重w_i，最终通过加权区域的特征值输出值$\tilde{a}$：

$$w_{i=}e^{a_i}（\sum_{j\in R} w_i \cdot a_i）$$

$$\tilde{a} = \sum_{i\in R} w_i \cdot a_i$$

这样，输出值便能实现池化内核所有的激活因子的甲醛求和，最终得到一个相对妥帖的池化结果。相较于最大池法，软池法没有那么极端的权重分配，能够更好的传递特征值，但是计算相对复杂一点，在第一步软池化完之后，即可继续进行后续卷积及最大池化。

对于输入特征图X，最大池化操作将其划分为$m \cdot m$的区域，然后在每个区域中选择最大的值作为输出值。具体来说，假设输入特征图的大小为：$H \cdot W \cdot C$则最大池化操作的输出特征图的大小为:

$$\frac{H}{m} \cdot \frac{W}{m} \cdot C$$

其中m为池化窗口的大小。

进一步来说，对于输入特征图的第i个区域，其对应的输出值为输入特征图中该区域内的最大值。最大池化操作的数学表达式可以表示为：

$$y_{i,j,k} = \max_{i'j'\in R_{i,j}} x_{i'j'k}$$

其中，$x_{i'j'k}$表示输入特征图的第i'行、第j'列和第k个通道的像素值，y_{ijk}表示输出特征图的第i行、第j列和第k个通道的像素值，$R_{i,j}$表示输入特征图中第i行、第j列的区域。

本模型的池选择的大小为 2×2，相对较小的池能减缓数据过快变小，最大池化就是指将 2×2 范围内的最大值保留下来，最终输出的尺寸变为原来的一半，即 127×127。第二，第三个池化层同样是 2×2 的矩阵，输出形状分别为 62×62，30×30。

在完成最大池化后，步长为 n 的特征图被压缩到 n/2 大小，在尽可能减少损失过快的同时降低了特征图尺寸，并保留了特征图中的重要信息，压缩了网络信息。这一步骤同样也避免了重复卷积导致模型过拟合。同时也保证了模型的生存性，能够更好的处理测试集中的内容，增强模型的鲁棒性。

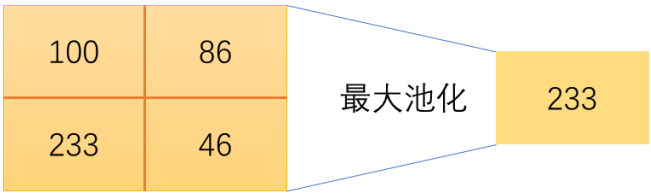

图 7
池化层原理图

5.2.5. 全连接层

全连接层是神经网络中的一种基本层类型，也被称为密集连接层或全连接层。它的主要作用是是整合前面几个卷积和池化数据与当前层的所有神经元相连，形成一个完全连接的网络。在全连接层中，每个输入都与每个输出相连，这使得全连接层能够学习输入和输出之间的任意复杂关系。

具体来说，全连接层中的每个神经元都与前一层中的每个神经元相连，并且每个连接都有一个相关的权重。这些权重是通过反向传播算法来学习的，目的是使得网络的输出尽可能地接近期望的输出。

在训练过程中，全连接层的输入是前一层的输出向量，输出是当前层的输出向量。在数据传入全连接层前，需要将三维的数据降维到一维，输出数据形状为 30×30 乘 128=115200。全连接层的每个神经元都将前一层的所有输出向量加权求和，并将结果传递给激活函数进行非线性变换，以产生当前层的输出向量。

全连接层的数学表达式可以表示为：

$$y_{i,j} = \sum_{k=1}^{n} w_{i,k} x_{k,j} + b_j$$

其中，$x_{k,j}$表示输入层第k个神经元的输出值，$w_{i,k}$表示对应位置的神经元的权重，而$y_{i,j}$则表示神经元在对应位置的输出，b_j表示第j个位置的偏置值。

最终，上述公式所得的数据在 sigmoid 函数的分类下，我们得以将路面图像分类成坑洼和平坦两类。

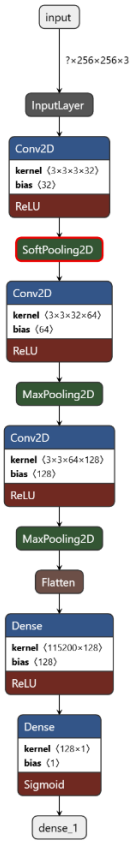

图 8
整体过程

6. 问题 2：对路面坑洼识别卷积神经网络模型的多维度评价

6.1. 对于整体模型选择的评价思考

卷积神经网络模型是近十年间才兴盛起来的模型，已经目前最常用的图像识别模型之一，它可以通过卷积操作和池化操作提取图像特征，然后通过全连接层进行分类。CNN 的优点是可以自动学习图像特征，适用于不同类型的图像数据，但需要大量的训练数据和计算资源。

由于参数较少，简单 CNN 模型的模型复杂度相对较低，难以捕捉到更复杂的图像结构和特征。简单 CNN 模型在处理一些复杂的图像分类任务时，可能需要采用一些数据增强技术来提高模型的性能。简单 CNN 模型在处理大规模数据集时，也可能会出现过拟合等问题，需要采用一些正则化技术来解决。

但本模型只包含几个卷积层和池化层，参数较少，容易实现和调试。训练速度也相对较快。同时在一定程度上能够提取出输入图像中的局部特征，从而实现图像分类、目标检测等任务。

除了卷积神经网络模型外，经典的机器学习算法支持向量机（SVM）在此前曾是本类问题的最优解，它可以将高维特征空间映射到一个低维空间，从而实现分类。

SVM 的优点是具有较好的泛化能力，但需要对特征进行手动设计，且对于图像数据的处理需要进行特征提取，而且这种提取难度极高，需要大量的时间尽力。同样，在 2012 年深度卷积神经网络 AlexNet 在 ImageNet LSVRC-2012 图像分类比赛上击败了支持向量机，侧面印证了 CNN 算法对此类问题的适应性。[2]

其他的诸如决策树、深度学习模型等都可以用于解决此类问题，但存在更大的缺陷，不做考虑。

6.2. 卷积神经网络模型内部参数选择的考量

在卷积层中，设置了相对较小但数量较多的卷积核，可以在不增加网络参数的情况下增加网络的感受野，使网络能够更好地捕捉到输入数据中的细节和上下文信息。同时可以增加网络的特征表达能力，从而提高网络的分类和识别性能。但此举也会增加计算量，增加网络的训练难度和计算负担。如果这样的的卷积核数量过多，也可能会导致网络的过拟合现象，从而影响网络的泛化能力。

在卷积时，模型没有进行 padding 操作，虽然这不可避免的造成了图像边缘的失真，但考虑到采集的图像信息集中在图片靠近中央的位置，可以忽略其造成的影响，还能节省算力。

在池化层中，最大池法可以有效减少特征图的尺寸，从而减少网络中的参数数量和计算量，同时也可以提高网络的鲁棒性和泛化能力。最大池化层还可以帮助网络学习到输入图像的全局特征，因为它在滑动窗口时会考虑到整个窗口内的信息，而不是只关注局部的特征。然而，由于最大池化层只保留了输入特征图中的最大值，因此会丢失一些细节信息，从而降低网络的精度。此外，最大池化层也会导致输出特征图的尺寸减小，从而降低网络的感受野，可能会影响网络对于较大物体的识别能力。

6.3. 迭代训练结果

考虑到卷积神经网络需要大量的样本进行训练，所提供的训练集数量不够充裕，因此引入了部分外部训练集进行训练。

在对模型进行了多次训练之后，分别计算准确率，精准率，召回率，F1 分数，ROC 曲线下面积（AUC），混淆矩阵以及交叉熵损失。

准确率是指分类器对所有样本的分类结果正确的比例。计算公式为：

$$准确率 = (TP + TN)/(TP + FP + FN + TN)$$

其中 TP 表示真正例（分类器正确地将正例分类为正例的样本数），TN 表示真反例（分类器正确地将负例分类为负例的样本数），FP 表示假正例（分类器错误地将负例分类为正例的样本数），FN 表示假反例（分类器错误地将正例分类为负例的样本数）。

精准率（是指分类器正确地将正例分类为正例的比例。计算公式为：

$$精准率 = TP/(TP + FP)$$

其中 FP 表示假正例。

召回率是指分类器正确地将正例分类为正例的比例。计算公式为：

$$召回率 = TP/(TP + FN)$$

F1 分数是精准率和召回率的调和平均数，用于综合评价分类器的性能。计算公式为：

$$F1分数 = 2*精准率*召回率/(精准率 + 召回率)$$

这些指标可以用来评估分类器在不同类别上的分类效果，以及分类器的整体性能。通常来说，准确率越高，精准率和召回率越接近，F1 分数也越高，说明分类器的性能越好。

图 9

迭代训练中的训练损失+验证损失与训练准确率+验证准确率

ROC 曲线下面积是 ROC 曲线与坐标轴围成的面积，通常用 AUC 表示。AUC 的取值范围在 0.5 到 1 之间，AUC 越接近 1，表示分类器性能越好。当 AUC=0.5 时，表示分类器的性能与随机猜测相同，无法区分正例和反例。

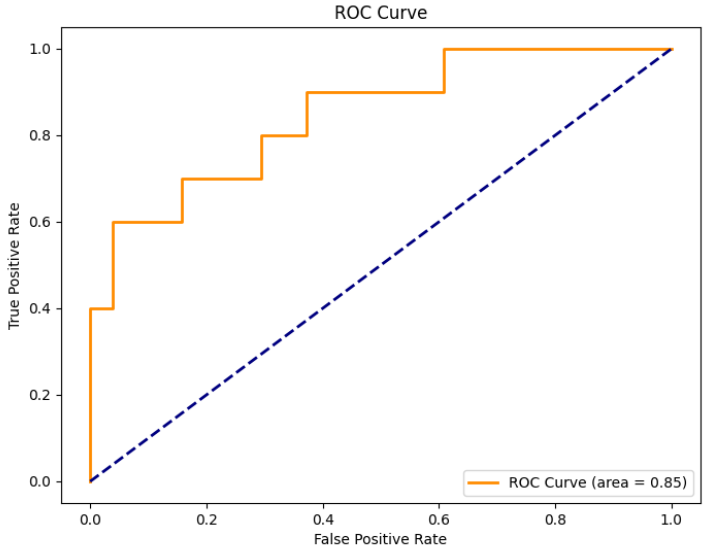

图 10

ROC 曲线下面积

混淆矩阵可以帮助我们理解分类器的分类效果，包括分类器的准确率、精确率、召回率等指标。例如，在一个二分类问题中，如果混淆矩阵的 TP 和 TN 值较高，而 FP 和 FN 值较低，则说明分类器的准确率和召回率较高，分类器的性能较好。反之，如果混淆矩阵的 TP 和 TN 值较低，而 FP 和 FN 值较高，则说明分类器的准确率和召回率较低，分类器的性能较差。

交叉熵损失函数的意义是：对于一个样本，如果模型的预测标签与真实标签相同，则该样本的交叉熵损失为 0，否则，损失值越大表示模型预测的不准确性越高在训练过

程中，我们通常使用反向传播算法计算模型参数的梯度，并使用梯度下降等优化算法来更新模型参数，以使交叉熵损失函数最小化。通过最小化交叉熵损失函数，模型可以逐渐学习到更准确的分类决策边界，从而提高分类性能。[3]

表 2 评价维度与参数

评价维度	数值
准确率	0.8361
精确率	0.6286
召回率	0.6
F1 分数	0.6921
ROC 曲线下面积（AUC）	0.8490
混淆矩阵	$\begin{pmatrix} 43 & 8 \\ 4 & 6 \end{pmatrix}$

根据给出的评估指标和混淆矩阵可以看出，分类器的性能比较好，准确率为 0.8361，AUC 值也比较高，说明分类器能够较好地区分正负样本。但是，精确率和召回率比较低，说明分类器在识别正样本和负样本时可能存在一定的误判和漏判，但影响不大。0.3637 张照片的运行时间:为 0.04115 秒，运行速度也相对较快。混淆矩阵中的数字可以进一步说明分类器的分类情况。

7. 问题 3：测试集识别结果

将所给测试集经过同样的预处理后输入模型，最后输出结果。由于某些图片的拍摄角度问题，我们对其进行了调整。测试结果显示，识别的平坦路面为 683 张，坑洼地面为 4259 张。详情参考上交的附件。

8. 参考文献

[1]佚名，卷积神经网络，卷积神经网络 - 维基百科，自由的百科全书 (wikipedia.org)

[2]大姨妈 V，【卷积神经网络发展历程】从 LeNet、AlexNet 到 ResNet、SENet，【卷积神经网络发展历程】从 LeNet、AlexNet 到 ResNet、SENet_深度学习网络发展历史 alexnet resnet-CSDN 博客

[3]IronmanJay，卷积神经网络（CNN）详细介绍及其原理详解，卷积神经网络（CNN）详细介绍及其原理详解_IronmanJay 的博客-CSDN 博客

附录

```
import os
import cv2
import numpy as np
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense, Conv2D, MaxPooling2D, Flatten


# 定义图像文件夹路径
image_folder = "C:\\Users\\ccclo\\Desktop\\10.28\\data\\OK"

# 定义数据和标签的列表
data = []
labels = []

# 遍历图像文件夹中的所有文件
for filename in os.listdir(image_folder):
    img_path = os.path.join(image_folder, filename)
    img = cv2.imread(img_path)
    if img is None:
        print(f"加载图像失败：{img_path}")
        continue  # 跳过该图像，继续处理下一个
    img = cv2.resize(img, (256, 256))
    data.append(img)

    # 创建标签
    if filename.startswith("normal"):
        labels.append(0)  # 正常路面标签为 0
    elif filename.startswith("potholes"):
        labels.append(1)  # 坑洼路面标签为 1

# 将数据和标签转换为 numpy 数组
data = np.array(data)
labels = np.array(labels)

# 划分数据集为训练集和测试集
X_train, X_test, y_train, y_test = train_test_split(data, labels, test_size=0.2,
random_state=42)

# 创建一个简单的 CNN 模型
model = Sequential()
model.add(Conv2D(32, (3, 3), activation='relu', input_shape=(256, 256, 3)))
model.add(MaxPooling2D((2, 2)))
model.add(Conv2D(64, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
```

```
model.add(Conv2D(128, (3, 3), activation='relu'))
model.add(MaxPooling2D((2, 2)))
model.add(Flatten())
model.add(Dense(128, activation='relu'))
model.add(Dense(1, activation='sigmoid')) # 输出层使用 sigmoid 激活函数，用于二分类

# 编译模型
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# 训练模型
model.fit(X_train, y_train, epochs=10, batch_size=32, validation_data=(X_test, y_test))

# 评估模型
accuracy = model.evaluate(X_test, y_test)[1]
print(f"测试集准确率: {accuracy * 100:.2f}%")

model.save("q3.h5")
```

第二段代码

```
import os
import cv2
import numpy as np
from keras.models import load_model


def predict_potholes(image_folder):
    # 定义数据和标签的列表
    data = []
    file_names = []

    # 遍历图像文件夹中的所有文件
    for filename in os.listdir(image_folder):
        img_path = os.path.join(image_folder, filename)
        img = cv2.imread(img_path)
        img = cv2.resize(img, (256, 256))  # 调整图像大小
        data.append(img)
        file_names.append(filename)

    # 将数据转换为 numpy 数组
    data = np.array(data)

    # 加载已经训练好的模型
    model = load_model("pothole_detection_model.h5")

    # 在输入数据上进行预测
    predictions = model.predict(data)
```

```
        predicted_labels = (predictions > 0.45).astype(int)  # 将模型输出转换为类别标签（0或 1）

        # 统计坑洼和非坑洼数量
        num_potholes = np.sum(predicted_labels)
        num_non_potholes = len(predicted_labels) - num_potholes

        # 输出预测结果
        results = []
        for i in range(len(predicted_labels)):
            if predicted_labels[i] == 1:
                results.append((file_names[i], "坑洼"))
            else:
                results.append((file_names[i], "平坦"))

        return results, num_potholes, num_non_potholes


    image_folder_path = ("C:\\Users\\ccclo\\Desktop\\10.28\\data\\try\\potholes")
    prediction_results, num_potholes, num_non_potholes = predict_potholes(image_folder_path)
    print("预测结果：")
    for filename, label in prediction_results:
        print(f"{filename}: {label}")

    print(f"坑洼数量: {num_potholes}")
    print(f"非坑洼数量: {num_non_potholes}")
```