中国农业大学

URP 项目结题报告

项目名称 基于视频监控的奶牛个体身份识别

学生负责人 颜君峻 学号 2017305010119

学　　院 信息与电气工程学院

联系电话 18141905669

项目成员 张靖祥、超博

导　　师 马钦

导师所在学院 信息与电气工程学院

导师联系电话 13391809180

基于视频监控的奶牛个体身份识别

摘要：随着“精准养殖”的兴起，奶牛的个体身份识别广受关注，是实现系统化、自动化对养殖个体针对性做出决策的第一步，具有深刻的研究价值和应用价值。目前奶牛场大多存在丰富的视频数据，但是限于条件无法得到充分的利用。随着人工智能领域的再次兴起，越来越多的高效率的目标检测和目标识别算法相继出现，但将其运用到奶牛养殖领域中的例子却很少。本项目针对奶牛群体中个体识别难的问题，研究基于视频监控的奶牛个体识别方法，主要包括制作奶牛身份数据集、奶牛个体目标检测、奶牛个体身份识别、搭建奶牛视频监控信息管理平台四个部分的内容。该平台能对未利用的奶牛监控资源进行加工利用，实现奶牛个体的身份识别。

在制作奶牛身份数据集环节，我们制定了详细的采集方案并与奶牛场联系，至奶牛场采集数据。拍摄牛背、牛侧身和牛头三个部分共 1250 图片数据、40 段视频数据，总大小 4.37GB。将拍摄的视频进行取帧，将分类好的奶牛图片进行清洗和标记。并分别将奶牛图片与标记产生的 json 文件一同进行翻转、放缩、平移、改变对比度等多种操作，进行数据的扩充，制作标准的奶牛身份数据集。

在奶牛目标检测环节，我们分别使用了经典的目标检测方法和基于 Faster RCNN 的目标检测方法。在经典的目标检测方法中，由于其局限性，只能对特定环境下的单只奶牛进行检测，鉴于其使用了颜色通道的方法进行分析，检测的效果不够理想。在基于 Faster RCNN 的目标检测方法中，我们改变了原有 Anchor box 的输入尺寸，用 TensorFlow 重新构建了整个检测部分的网络，并在 Proposal 部分提出了待检测目标的位置，实现了奶牛的检测。

在奶牛个体身份识别环节，我们分别用机器学习的传统方法 SVM、深度学习的深度神经网络 Alexnet 方法和 VGG 方法进行了实验。其中 SVM 识别准确率 71.4%，Alexnet 网络识别准确率 97.33%，VGG 网络识别准确率达 99.17%。实验结果表明，传统机器学习方法难区分奶牛场复杂环境下的前景和背景，识别准确率较低，无法运用到奶牛场复杂的实际环境中去；深度神经网络在复杂环境下仍具有较强的特征提取能力和身份识别能力，具有应用到复杂环境下的奶牛身份识别的能力。同时，研究表明加深网络卷积层和池化层的层数有利于提高识别准确率。

在搭建奶牛视频监控信息管理平台的环节中，我们开发了一款 B/S 架构的网站应用程序。选择使用 ASP.NET MVC 框架进行用户界面的开发，开发工具为 Microsoft Visual Studio。该平台提供了奶牛场监控视频存放接口、奶牛个体图片分类接口、基于深度学习的奶牛个体身份识别模型框架。用户使用该平台可以方便的构建自己的奶牛身份数据库，通过监控视频来识别奶牛身份，达到对奶牛的实时监控和个体定位的目的。该平台实现了奶牛场监控视频资源的统一管理和奶牛个体身份的实时监控，对奶牛资源进行精细的加工利用。

关键词：图像识别；神经网络；数据平台。

目录

1 项目背景

近年来随着“精准养殖”的兴起，奶牛的个体身份识别广受关注，是实现系统化、自动化对养殖个体针对性做出决策的第一步，具有深刻的研究价值和应用价值。基于 EPC（电子产品编码技术）和 RFID（射频识别技术）的物联网技术[1]是当前的主要识别手段，但有着成本高、设备要求先进、动物存在应激性等缺点。随着人工智能的再次兴起，基于图像识别的奶牛个体身份识别方案成了学者关注的对象。

卷积神经网络（convolutional neural networks，CNNs）[2]是基于深度学习理论的一种人工神经网络，由于该网络可直接将图像数据作为输入，避免了图像预处理和特征抽取等复杂操作。发展到现阶段的卷积神经网络已经被广泛使用在各个领域，在识别和决策中起到重要作用。将卷积神经网络应用到奶牛的个体识别上具有诸多优点，是大势所趋。

在早期研究中，Xia 等[3]提出了一种基于局部二值模式（local binary patterns，LBP）纹理特征的脸部描述模型，并使用主成分分析（principal component analysis，PCA）结合稀疏编码分类（sparse representation-based classifier，SRC）对奶牛脸部图像进行识别。该方法具有一定的准确性，但对识别的场景有诸多特殊要求，难以应用到实际场景中去。

注意到奶牛的区干的纹理特征，赵等[4]提出了一种基于视频跟踪的奶牛识别方法。将 LeNet-5 网络[5]运用到奶牛的个体识别，将每头奶牛视为一种模式，奶牛个体识别即对某一未知奶牛图像进行识别和匹配。结果表明具有良好的抗干扰能力，具有应用到复杂场景下的奶牛个体识别。但识别精度不高，有较大提升空间。

刘等[6]提出一种利用视频中每一帧图像建立高斯混合模型的奶牛定位跟踪算法。在此基础上生成奶牛个体侧身的彩色图像，利用迁移学习对 Alexnet[7]网络进行训练和微调，最终实现较高准确率的奶牛个体识别。遗憾的是难以识别多头奶牛，这使得识别的效率降低。

随着深度学习在目标检测上的开山之作 R-CNN[8]的诞生，William[8]等首次将目标检测算法应用到奶牛个体身份识别中去。利用无人机采集奶牛的背部图片，结合基于 VGG[10]的 R-CNN 网络对奶牛图片进行了定位和检测；同时使用长短期记忆网络（Long Short-Term Memory，LSTM）的方法对奶牛视频进行跟踪和检测。在视频检测方面具有较高的 mAP，但 FPS 值较低，达不到实时检测的效果。

在本项目中，我们将进一步把深度学习的手段应用到奶牛个体身份识别中去，采集并建立属于自己的奶牛身份数据集，分别对牛脸、牛侧身和牛背三个不同部位的数据集进行目标检测和目标识别。对多个成熟的深度学习模型进行了实验和研究，提出了我们的解决方案，设计了交互友好的软件界面。

2 数据集

2.1 数据集采集

数据采集于 2019 年 12 月 14 日下午 3 点-下午 5 点，北京市顺义区张镇张各庄村的三元绿荷奶牛养殖中心，天气晴朗，下图展示了项目成员在三元绿荷奶牛养殖中心进行数据采集的有关情况（图

2.1-1 A,B,C,D)。使用苹果手机 iPhone 7 的原生相机拍取相片和视频。原始图像大小：4032*3024 或 3024*3024，分辨率：72dpi，位深度：24，曝光时间：1/339s，格式：JPG。原始视频大小为：640*480，平均 FPS：27.8，格式：MP4。原始数据集共 4.37GB，其中图片约为 1250 张，视频约 40 段。

按照拍摄到奶牛身体的不同部位，可以将原始数据划分为牛背、牛侧身和牛头三个部分。其中，背图片约为 400 张，视频约为 10 段；牛侧身图片约为 700 张，视频约为 20 段；牛头图片约为 150 张，视频约为 10 段。

图 2.1-1A,B,C,D 采集数据现场图

2.2 数据集制作

在深度学习中，数据集的质量对模型有着至关重要的影响[11]，现阶段在这方面有丰富的研究成果，给数据集的制作提供了大量的理论依据。采集到的原始数据存在着数据不完整、数据不一致和样本不均衡等问题，不能直接拿来使用。为了得到有利于训练模型的数据集，我们将对视频数据进行取帧，对图像数据进行清洗和过采样等操作。同时我们将采用数据增强的方法来增加训练的数据量，提高模型的泛化能力；添加部分噪声数据，提升模型的鲁棒性，防止过拟合。

2.2.1 数据预处理与标记

数据清洗(Data cleaning)作为数据处理的第一个环节，目的在于删除重复信息、纠正存在的错误，并保证数据一致性。在本项目中，对于上述的 87 个分类的奶牛侧面图片，我们进行了初步筛选分类，并分出了 47 组侧面图片完整，清晰且比例适当的奶牛图片作为目标识别使用。挑选出 67 张奶牛数量与位置适中的图片进行目标检测使用。

对于奶牛背部的图片的检测与分类，目前已经有较为成熟的基于 RCNN 网络的 Holstein Friesian Cattle 奶牛的识别方法[9]。而对于奶牛的头部和牛侧身的图片，目前检测的方法比较少，因为奶牛的头部和侧面的拍摄角度更加灵活多变。即便同一头奶牛的不同侧面，拍摄的图像也是截然不同的，这给深度学习带来了极大的挑战。在本项目中，我们仅仅选出了奶牛一个侧面的图像进行图像的识别训练识别。选出了奶牛群体的图片并标记了多头奶牛的位置，用作目标检测。

完成数据清洗后，我们将图片统一调整为 1000*600 像素的尺寸，并用标记软件 LabelMe 对其进行了标记。在每个图片中，我们标记出来每头奶牛的位置，并分为两组。第一组的标签为 cow，表明是一头奶牛，但是不对其进行分类（下图 2.2.1-1 黄色方框）。第二组标记为 cow+奶牛编号（如下图 2.2.1-1 绿色方框）。这两类都作为目标检测的实验使用，第二组作为目标识别使用。

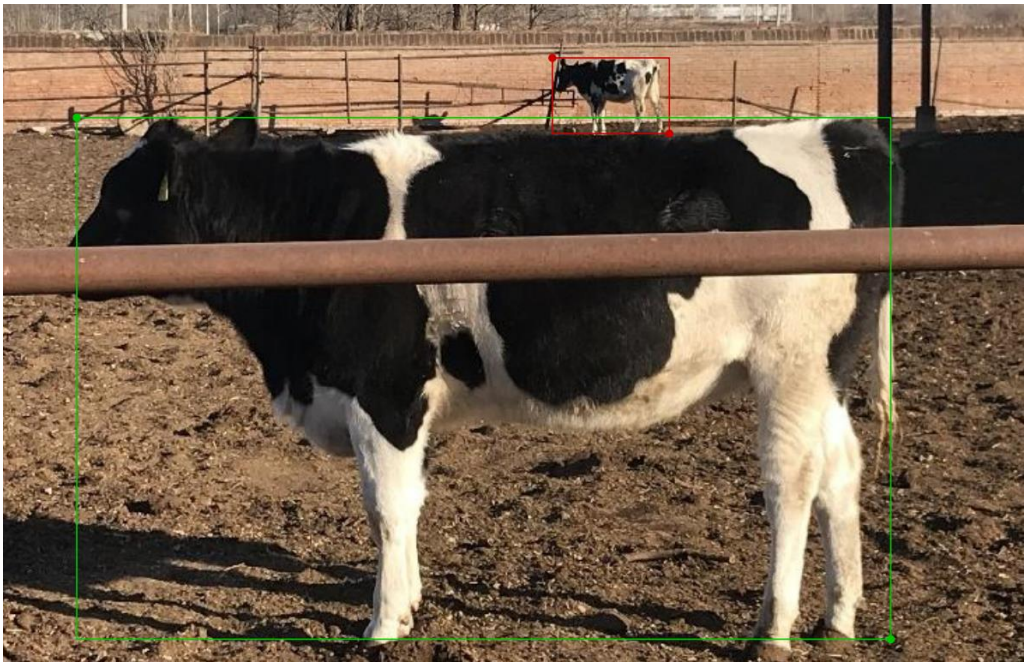

图 2.2.1-1 标记示例图

2.2.2 数据扩充

在数据标记过程中，图片经 LabelMe 编码后统一存储在了 Json 文件中。我们用 python 读取 Json 数据，用 LabelMe 将编码后的图片解码为 1000*600*3 的图片矩阵。用 OpenCV 对图片进行系列变换，用 Numpy 对标记框的位置进行变换。最后，重新将图片编码为 LabelMe 格式，并将其保存。我们将上述方法分装为 python 类，实现了对由 LabelMe 保存的 Json 文件进行自定义的图片扩充。

我们在众多的数据扩充的方法中选取平移，水平翻转，改变对比度和亮度，放缩这四种方式对数据集进行扩充。原始的 CNN 网络对于图像特征的提取不具有平移不变性[7]，对于步长大于 1 的下采样，均会导致平移不变性的丢失。即使是移动一个像素，池化层的结果也会发生巨大的变化，这严重影响了网络的鲁棒性。因此，即便是特征提取出了奶牛的位置，也难免存在像素级别的偏差，而这微小的偏差会对结果产生巨大的影响。所以平移操作对于数据的扩充是十分有必要的。改变对比度是为了适应在一天的不同时间，不同天气的光照条件下能对奶牛进行识别。放缩则是为了适应奶牛处于不同的距离时都能够进行识别。处理结果如下图所示（图 2.2.2-1）。

图 2.2.2-1 数据增强结果图

在上图（图 2.2.2-1）所示的图片中，我们对每张图片进行了水平翻转操作。放缩操作比例分别为 0.7、1、1.2；平移操作分别为向左平移 100 像素、不平移和向下平移 100 像素；像素每个通道的值降低为 0.7 倍减 20、不变和 1.2 倍加 20。平移和放缩操作中，边界部分用 0 填充；在亮度值的操作中，超出 0 或 255 边界的值记为边界值。需要注意的是，在显示图片的时候，matplotlib 使用的颜色值为 RGB，而 LabelMe 使用的颜色为 BGR，因此需要用 OpenCV 对其进行相应的转化。

2.2.3 视频取帧

本项目是基于视频监控的奶牛个体识别，在处理视频中取帧是必不可少的步骤。我们使用 FFmpeg 对视频进行图片截取，对截取出的图片进行检测。取帧效果如下图所示（图 2.2.3-1）。我们对视频中的图片每秒截取一张图片，并保存为 640*480 像素的图像。由于标记数据所使用的图片像素为 1000*6000，截取这些图片需要进行从新调整大小以适应网络的输入尺寸。

图 2.2.3-1 视频取帧效果图

3 目标检测

相比身份识别，目标检测的难度更高。我们不仅要判断图片奶牛是哪头奶牛，还要在图片中标记出它的位置，图片中的奶牛可以是多头。基于深度学习的目标检测算法 Faster-RCNN[12]，在卷积神经网络的特征提取之后，对损失函数进行重新定义，以实现减小识别误差的梯度下降。其实现的方法十分复杂，从原理上讲不适合用 TensorFlow 这种数据流图的平台进行处理。为了完成网络的各个部分计算，并进行反向传播，最终我们选择使用结合经典方法和自定义损失函数的 VGG 网络进行目标检测。

3.1 经典的理论方法

3.1.1 实现流程

在奶牛图片中，奶牛与背景存在明显差异。牛场的背景为黄色的地面和蓝色的天空，而奶牛则为白色和黑色，这使得经典的按颜色通道提取的方法得以实现。我们将白色和黑色的区域提取出来，并进行一系列的处理，找到一个合适的目标区域。这种方式的缺点就是在有较暗的环境下难以区分奶牛与天空的颜色，并且只适用于图片中只有一头奶牛的情况。我们具体的实现流程如下图所示（图 3.1.1-1）

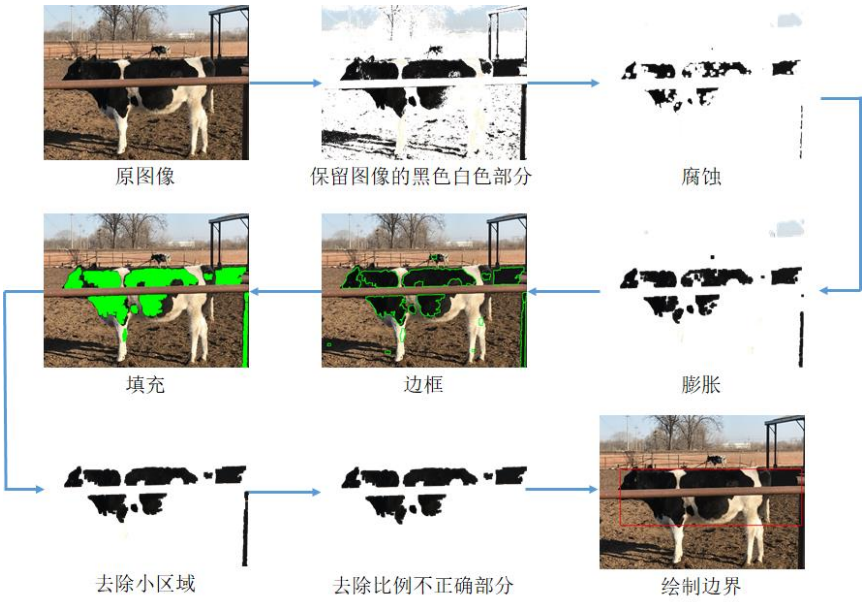

图 3.1.1-1 流程图

我们先将原图像的 RGB 三个通道分开，利用 Numpy 矩阵运算功能分别对三个通道的矩阵进行比较判断，将通道颜色值小于 30 的点（黑色）和大于 220 的点（白色）记录为 True，其余记录为 False。然后对矩阵进行按位与操作，将三个通道的矩阵进行合并，得到 alpha 透明度通道，即：只有三个颜色通道的值都满足条件的情况下才为不透明，否则其余的像素点都为透明。最后使用 OpenCV 将四个通道合并，并保存为 PNG 格式的图像，由此得到了流程中的第二个图像。使用 PNG 格式而非 JPG 格式的图片是因为 JPG 格式图像不支持透明通道。

然后我们对透明度通道进行了矩形内核形状（大小为 10*10）的腐蚀操作，得到了第三步的图像；接下来又用了相同的内核进行了膨胀操作，得到了第四步的图像。上述操作可以过滤掉绝大部分图像中的噪声，得到近 20 个连通区域。但是从绘制的边框结果来看，仍然有不少的非奶牛区域被框选。

之后我们利用 OpenCV 中计算边框面积的方法，计算出所有边框围成区域的面积，去掉以像素为单位的面积小于 500 的区域，剩下大约 12 个区域。由于奶牛侧身花纹的形状不会出现右侧电线杆的细长形状，我们可以将剩下的区域中高度宽度比大于 2 的全部略去，最后剩下近 8 个区域。我们假定这 8 个区域都是同一头奶牛身上的花纹，就找到了 8 个区域的边界进行边框绘制，得到最后的结果。

3.1.2 结果分析

从直观角度上讲，这种方式不适用于多头奶牛的情况，并且容易受到多种因素的干扰，比如在这个例子中，由于出现了阴影，错误的将阴影也算做了奶牛身体的一部分。在此过程中，颜色值范围、腐蚀与膨胀核的大小、去除小区域的面积大小、去掉错误的长宽比例，这四个范围的设定都将影响框线的结果。设定的条件宽松，将会框选出多余的区域；设定的条件严格，将会框选出过小的区域。最终上述的四组参数的设定都是经过试验得到的。

3.2 基于 Faster RCNN 的检测方法

Yolo[13]和 Faster RCNN[12]这两大经典基于深度学习的算法，作为端对端的识别模式，这两大算法无论从速度还是准确率方面都远高于同期的其他算法。虽然 Faster RCNN 结构较为复杂，复现起来难度较大，但是其思想仍然值得我们学习。本文中我们采用了基于 Faster RCNN 思想的目标检测方法。

VGG 作为目标识别网络，其主要功能是提供特征提取功能，而真正的目标分类则是在最后的全连接层。我们学习借鉴了 Faster RCNN 的思想，去掉了 VGG 网络的全连接层，并缩小了网络规模，利用其特征提取的作用，提取我们用于下一步操作的特征图。随后我们计算了和 Faster RCNN 中一样比例的 Anchor box，但大小略作调整以适合奶牛的尺寸。与其不同的是：1.我们没有去除超出边界范围的边框，因为我们期待可以通过定义适当的损失函数，将边框变化到正确范围内。2. 我们在训练阶段没有使用 NMS（非极大值抑制）的方法，因为我们的奶牛标记数据中奶牛的标签数量较少，进行 NMS 后将会导致剩余的可选 Anchor box 数量非常少，以至于无法完成训练。

3.2.1 网络的细节与训练

网络的整体架构定义如下图（图 3.2.1-1）所示。其中，输入为两个参数，一张图片和图片上目标边框的位置。图片的输入尺寸为 320*400*3，边框的输入形状为 5*4，即：最多出现 5 个边框，每个边框给出四条边的 X 或 Y 轴的坐标。

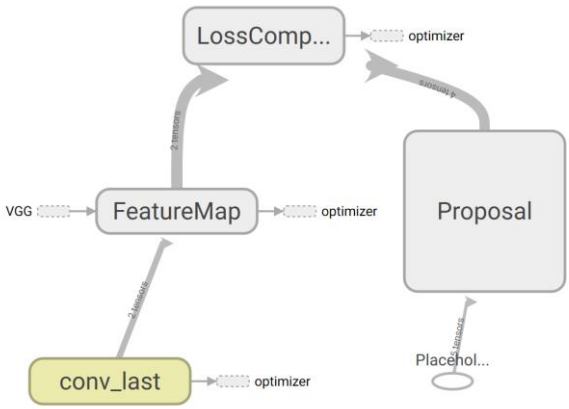

图 3.2.1-1 网络整体架构图

图片输入直接进入上图（图 3.2.1-1）左边的 VGG 卷积层进行特征提取，具体的卷积层架构如下图所示（图 3.2.1-2），一共经过 3 个池化层和 9 个卷积层，均使用 ReLU 作为激活函数，卷积层的输出值即是 Feature Map 的输入值，40*50*150 的特征值矩阵。

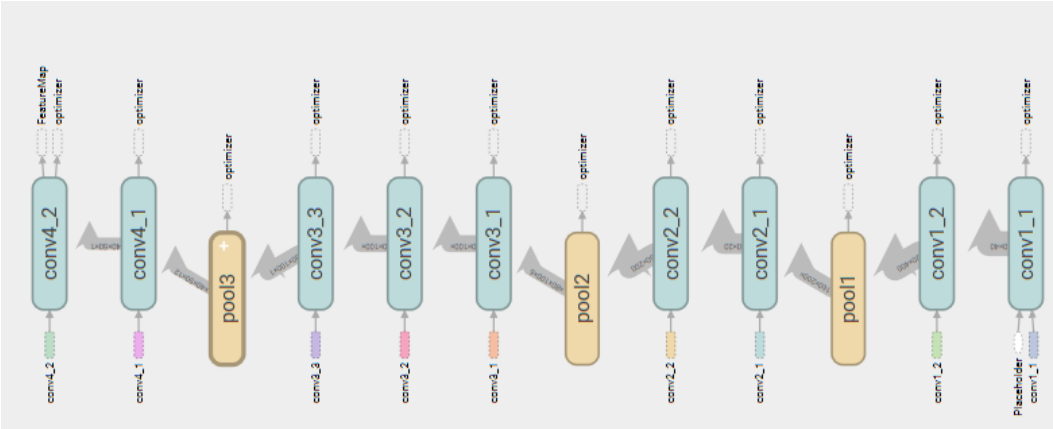

图 3.2.1-2 特征提取网络架构图：基于 VGG16

在 Feature Map 这一个环节，我们对输入值做了一个 1*1 的卷积，并将输出结果 40*50*54 的矩阵分割成 40*50*9*2 与 40*50*9*4 这两组，具体的原理与 Faster RCNN 一致，即：前者用于 SoftMax 分类，判断特征点是否为奶牛，后者用于 Anchor box 边框的微调。9 个 Anchor box 的长度和宽度我们设计了更加符合奶牛尺寸的大小，具体参数见下表（表 3.2.1-1）

表 3.2.1-1 Anchor box 的尺寸

编号	长度/像素	宽度/像素
1	64	64
2	45	90
3	90	45
4	128	128
5	90	180
6	180	90
7	256	256
8	180	360
9	360	180

在这一个环节我们一共产生了 40*50*9=18000 个 Anchor box，接下来我们要对这些 Anchor box 进行挑选。不同于 Faster RCNN 论文中的，我们并没有将超出图片范围的边框筛除，而是保留下来，期待如果在 Proposal 的过程中，如果选中了超出范围的边框，可以按照正常的流程去减小损失函数。而这样做的好处在于，如果待检测目标恰好在角落，将会被一视同仁的检测到并进行回归。而对 Proposal 阶段（下图 3.2.1-3），我们先产生了 18000 个 Anchor box，然后将其与输入的 10 个边框进行比较。由于 TensorFlow 只限于固定形状的矩阵计算，所以当输入的边框小于 10 的时候，我们要对其进行以 0 进行填充。而在选择的时候也必须要进行 10 个循环的选择（因为与 0 做选择一定为 False，所以以 0 填充不会影响结果，具体的计算过程见下）。

在 intersect Box 的阶段我们将输入的边框与基准的 18000 个 Anchor box 进行求 IOU 的运算。一共进行 10 次循环，具体原因见上。首先我们开辟了相同大小的 0 填充的大小为 40*50*9 的 IOU 结果矩阵和用 0 填充的大小为 40*50*9 的 dx，dy，dw，dh 偏移量矩阵。期待输出结果为 40*50*9*4 的偏移量矩阵用于计算损失函数，和 40*50*9 的 IOU 矩阵用于调出正样本和负样本进行学习。

在每次循环中我们将边框矩阵 tensor 切片出一组 x、y、w、h，先对每一个输入的 Anchor box 进行判断是否相交，结果转化为 float32 型的 tensor 张量。再计算 IOU 矩阵，并将 IOU 矩阵与前者相乘，得到的即是正确的相交比例。然后将其与带输出的 IOU 矩阵进行求最大操作，期待得到与图像中某个边框得到最大的 IOU 值。将扩充的 IOU 部分记录的序号记录下来，并与本次计算所得到的 dx，dy，dw，dh 相乘，去除所有的没有本次循环得到的结果。并将这四个结果加到待输出的 dx，dy，dw，dh 中，最后合并到一个矩阵里进行输出。

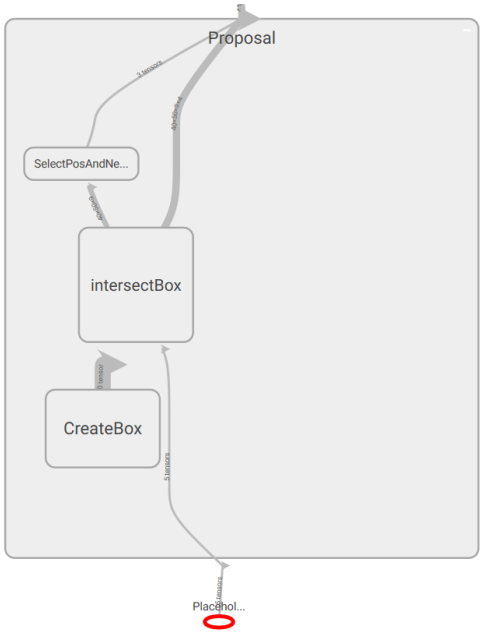

图 3.2.1-3 Proposal 阶段示意图

我们将计算好的 IOU 矩阵进行挑选正负样本的操作。一共挑选出 100 个正样本和 100 个负样本。正样本从 IOU 排序所得的前 200 的结果中随机挑选 100 个，记录下标。负样本为 IOU 排序结果小于 0.3 的数中进行排序，得到结果的前 5000 个中随机挑选 100 个，并记录下标（注：下标范围是 0~17999）。将挑选出的正负样本下标作为 Proposal，与 dx，dy，dw，dh 一起返回，传入到损失函数的计算中。

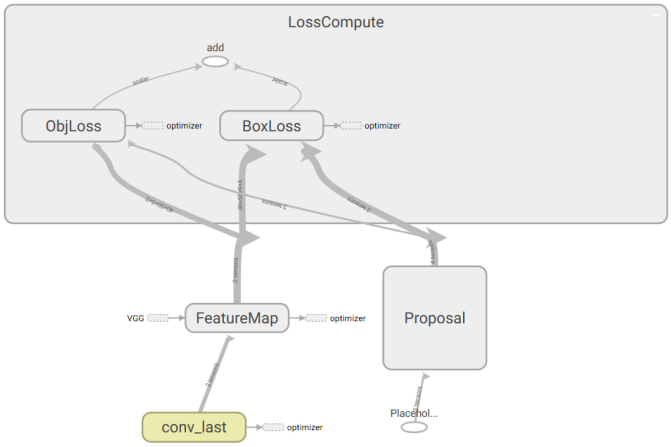

图 3.2.1-4

上图（图 3.2.1-4）中为损失函数的计算。损失函数包括两部分（部分与 Faster RCNN 一样），分别为是否为物体的损失和边界的损失。公式如下，对于偏移量 Box Loss 我们只计算正样本的损失，希望其边框更接近正确值，因为在检测的过程中，我们期待的是从 Proposal 这里提出的就是与目标接近的 Box，而我们只需要对这部分减小损失就可以了。Proposal 提出的负样本是我们不需要计算偏移量的，以为检测的过程中就已经被 Proposal 排除了。对于检测是否为物体的 Obj Loss，我们则希望更准确一些，希望负样本的值更低，正样本更高。

$$\begin{cases} objLoss = \dfrac{1}{200}\sum_{i=0}^{200} F_{cls}(p_i, p_i^*) \\ BoxLoss = \dfrac{1}{100}\sum_{i=0}^{100} F_{reg}(t_i, t_i^*)/4 \\ sumLoss = objLoss + BoxLoss \end{cases}$$

其中：$F_{cls}(p_i, p_i^*)$ 为p_i与p_i^*的交叉熵损失，这两个都是经过 Proposal 下标取出的 200 个正样本和负样本的 200*2 的矩阵。$F_{reg}(t_i, t_i^*)$ 则为 Proposal 部分输出 40*50*9*4 部分的 dx，dy，dw，dh 与 Feature map 输出的相应部分按下标取出对应元素之后的值的计算，t_i为 Proposal 输出的真实偏差，而t_i^*则为 Feature map 输出的偏差。这两个值均为 100*4 的矩阵。分别计算其相应元素之差，再取绝对值并求和。

3.2.2 网络训练

在数据方面，我们标记了 70 头带有奶牛位置数据的图像，并将其分别按照平移[0,0], [100,0], [0,100], [-100,0], [0,-100], [50,50], [50,0], [0,50], [-50,0], [0,-50]的方式进行平移扩增，分别放缩了 0.7、1.0、1.2 倍，并进行了左右翻转，对比度变为 0.6、1.0、1.3 这三种扩增模式。每张图片扩增了 10*3*2*3=180 倍。取出前 56 头奶牛扩增为了 10800 张图像作为训练数据，后 14 张扩增为了 2520 张作为检测数据。

在训练的过程中，我们每次只训练 16 张图片，并设置了 0.0001 的学习率。对 10800 张图片一共进行了 30 个周期的训练。受限于硬件平台的限制，我们并没有训练太多的周期。其中网络的总体损失如下图所示（图 3.2.2-1）。

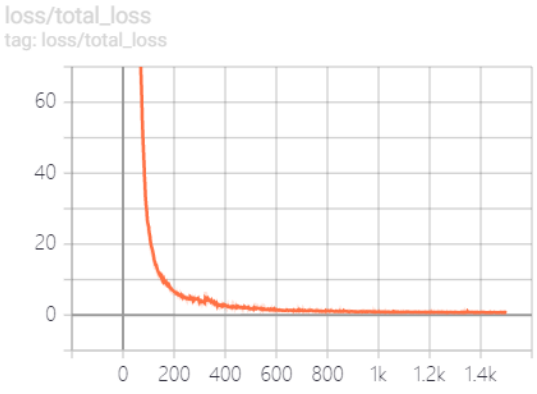

图 3.2.2-1 损失函数变化图

3.2.3 目标检测

在目标检测的过程中，我们将图片输入训练完成的网路中去，获取一个 40*50*9*2 的是否为物体的检测结果和一个 40*50*9*4 的偏移量结果。我们将偏移量 dx，dy，dw，dh 四个值分别与训练时候的 Anchor box 进行计算（公式如下），得到的结果与判断是否为物体的结果一同进行 reshape，转化成 18000*6 的矩阵。先对其进行了裁剪，整个矩形都超出范围的直接删除，部分超出边界的按照边框的位置进行裁剪。

$$\begin{cases} xcenter = AnchorX + AnchorW \times dx/2 \\ ycenter = AnchorY + AnchorH \times dy/2 \\ w = e^{dw} \times AnchorW \\ h = e^{dh} \times AnchorH \end{cases}$$

然后对概率进行计算，取$\text{objPro} = (\text{obj}[0] - 0.5) \times 2$，当 obj[0]大于 0.65 时才进行计算，裁减掉小于 0.65 的结果。最后将结果进行非极大值抑制，IOU 的边界设置为 0.5，因为在我们的图片中，奶牛重叠的概率很小。最后按照可能性的顺序绘制图像。最后的识别结果见下图（图 3.2.3-1），这是一张模拟奶牛在黑夜的图片的检测结果。

图 3.2.3-1 检测结果示意图

4 身份识别

我们将目标检测与身份识别区分开来，奶牛的目标识别算法可以归结为分类问题。分类问题有很多种解决办法，传统机器学习的方法包括决策树、贝叶斯、支持向量机、逻辑回归等，该方法对背景复杂的奶牛图片分类效果不佳。深度学习方法中以深度卷积神经网络为主要内容。在本项目中，我们对机器学习的方法、和深度学习的方法都进行了实验，挑选出几个具有代表性的实验结果进行阐述。

4.1 SVM 方法

SVM 方法是经典的机器学习算法，可以应用到分类任务。SVM 的核函数方法可将非线性可分的样本升至高维空间，从而线性可分。在许多小规模、特征数较少的分类问题上有着广泛的应用。遗憾的是，实验表明，奶牛场养殖环境复杂，背景和前景的区分度不过高，使得 SVM 分类难以选择到奶牛合适的特征，故识别准确率较低，难以应用到实际的奶牛场奶牛个体身份识别上去。

4.1.1 特征工程

SVM 不具备像卷积神经网络一样的端对端识别的能力，在训练之前需要手动选取特征进行分类。对 202 张图片，按训练集与测试集 9:1 进行划分，即训练集 180 张图片，测试集 22 张图片。对于每张图片，我们选取图片灰度共生矩阵的特征上（ASM）、对比度、能量、R 通道均值、G 通道均值、B 通道均值、色相均值、饱和度均值、亮度均值共九个方面的特征。

分别选取线性核函数 linear、多项式核函数 ploy、径向基核函数 rbf 和 sigmoid 四个核函数进行 SVM 奶牛分类器的训练，使用网格搜索的方法进行参数的选取。其中所有核函数均对参数“惩罚因子 C”从 0.0001 搜索至 10000，每次搜索扩大 10 倍。多项式核函数对“多项式维度 degree”从 1 搜索至 10，步长为 1；径向基核函数的“γ 参数”在 0.1, 0.2, 0.4, 0.6, 0.8, 1.6, 3.2, 6.4, 12.8 的倒数中选取；sigmoid 核函数对“常数项 coef0”从 0.1 搜索至 1，步长为 0.1。

4.1.2 结果分析

上文中选取的参数调整结果为：当 C=0.1 时，线性核函数结果达到最佳，为 71.43%；当 C=10，degree=2 时，多项式核函数结果达到最佳，为 66.43%；当 C=0.0001，γ=2.5 时，径向基核函数分类效果达到最佳，为 9.52%；当 C=0.001，coef0=0.0 时，sigmoid 核函数达到最佳，为 30.16%。综上所述，选取线性核函数，当惩罚因子 C=0.1 时，识别准确率为 71.43%，达到最高。下表（4.1.2-1）展示了详细的结果，其中运行时间是指在网格搜索最优参数的过程中消耗的时间。运行环境为 Baidu AI Studio（CPU：8 核，RAM：32GB）。

表 4.1.2-1 SVM 结果对比表

	线性linear	多项式ploy	径向基rbf	sigmoid
准确率	71.43%	66.43%	9.52%	30.16%
运行时间	3.86s	9.96s	0.33s	0.20s

4.2 Alexnet 方法

Alexnet[7]是深度神经网络的开山之作，虽然网络结构较为陈旧，但不乏一些经典的原理和思想。用奶牛数据集来训练 Alexnet 神经网络模型，是测试奶牛分类是否适合使用深度神经网络的重要手段。实验表明，深度神经网络能很好的胜任奶牛个体分类的任务，具有较高的准确率和鲁棒性。

4.2.1 模型搭建

采用经典的 Alexnet 建议架构作为实验的深度网络模型。共八层神经网络，前五层为卷积层（conv1-conv5），后三层为全连接层（fc6-fc8），其中最后一层（fc8）为具有 1000 个输出项的 SoftMax。在 conv1 和 conv2 之后跟着正则化层 norm1 和 norm2，在每层网络的输出部分设有激活函数 ReLU。在 norm1 和 norm2 以及 conv5 之后设有最大池化层 pool1，pool2 和 pool5。在最后两个全连接层有随机失活 Dropout 操作。详细结构如下图所示（图 4.2.1-1）。

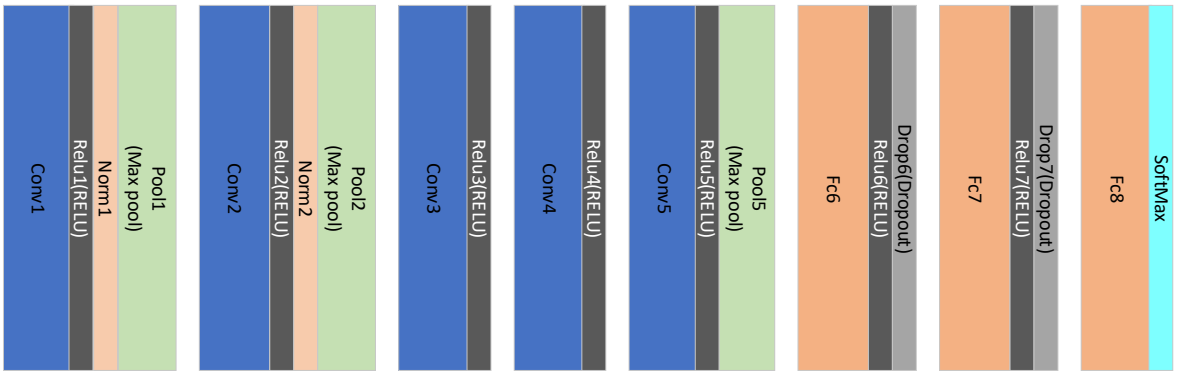

图 4.2.1-1 Alexnet 网络架构图

4.2.2 训练方法

从头开始训练一个神经网络需要大量的图片数据并迭代较长的时间，为了缩短实验周期同时防止由于训练样本不足而带来的准确率下降的问题，我们采用迁移训练的方法。加载已训练过的 Alexnet 权重，该权重在 ImageNet 上进行了充分的迭代。我们冻结网络的前七层，对最后一层 fc8 进行迁移训练。对于 Alexnet 这样层数不深的网络，只对 fc8 进行迁移训练就能得到很好的效果，假使对 fc7 和 fc6 也做迁移训练，效果并不会有明显提升，甚至会有略微下降[8]。

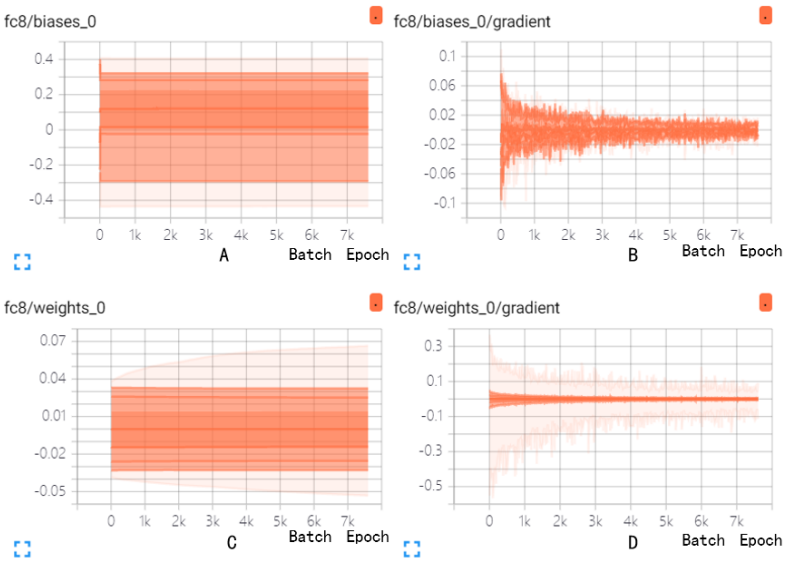

图 4.2.2-1A B C D fc8 权重梯度分布图

上图（图 4.2.2-1A B C D）分别展示了全连接层 fc8 的偏置分布，全连接层 fc8 的偏置的梯度分布，以及全连接层 fc8 的权重的分布和全连接层 fc8 的权重的梯度分布。由于迁移训练冻结前面七层，只对最后一层做迁移训练，所以我们只可视化了 fc8 的有关信息。可以发现训练过程中的偏置和权重变化范围幅度不大，但梯度变化较大，说明初期模型虽然不能拟合数据集，但是仍有较强的特征提取能力。

下图（图 4.2.2-2A B C D）展示了全连接层 fc8 的偏置变化情况，全连接层 fc8 的偏置的梯度变化情况，以及全连接层 fc8 的权重的变化情况和全连接层 fc8 的权重的梯度变化情况。权重的变化较为平缓，而偏置的变化较为复杂。通过分析此图可以得到的一个事实是：迁移训练过程中，偏置比权重对新数据更为敏感，变化幅度和范围更大。

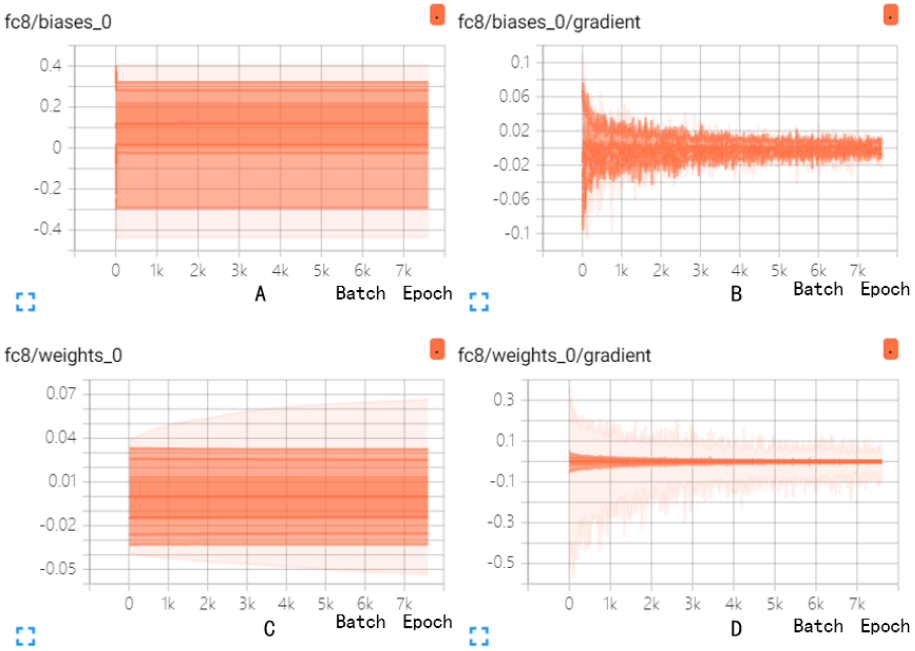

图 4.2.2-2A B C D fc8 权重梯度变化图

4. 2. 3　结果分析

我们的奶牛侧身数据集（原始 202 张，扩增 52 倍，其中训练集 9453 张，测试集 1050 张。）在 Google Colab（GPU：Tesla K80 13GB）上以 0.001 的学习速率、128 的批处理大小下迭代 100 次，训练时长 5881.689 秒，测试单张图片时长约为 0.02 秒。下表（表 4.2.2-1）展示了模型在训练集、测试集和原始图片集上的准确率，其中训练集准确度为 98.61%、测试集准确率为 97.33%、平均准确率为 98.48%、原始数据集准确率达 100%。

表 4.2.3-1 Alexnet 网络检测准确率

训练集	测试集	训练集+测试集	原始数据集
98.10%	97.33%	98.48%	100%

下图（图 4.2.3-1A B）展示了迭代过程中准确率的上升情况和损失函数的下降情况，横坐标为 batch 的次数，纵坐标为准确率或损失函数值。可以看到在训练过程中起伏较小，说明很少出现陷入局部最优解的情况，收敛过程较为顺利。有研究表明，预训练好的模型权重具有较强的特征提取能力，迁移训练出来的网络更具鲁棒性。由于该模型采用的初始权重在包含奶牛图片的 ImageNet 上进行过训练，本身对奶牛个体的信息较为敏感，因此在迁移训练中，损失函数得以较快的收敛。

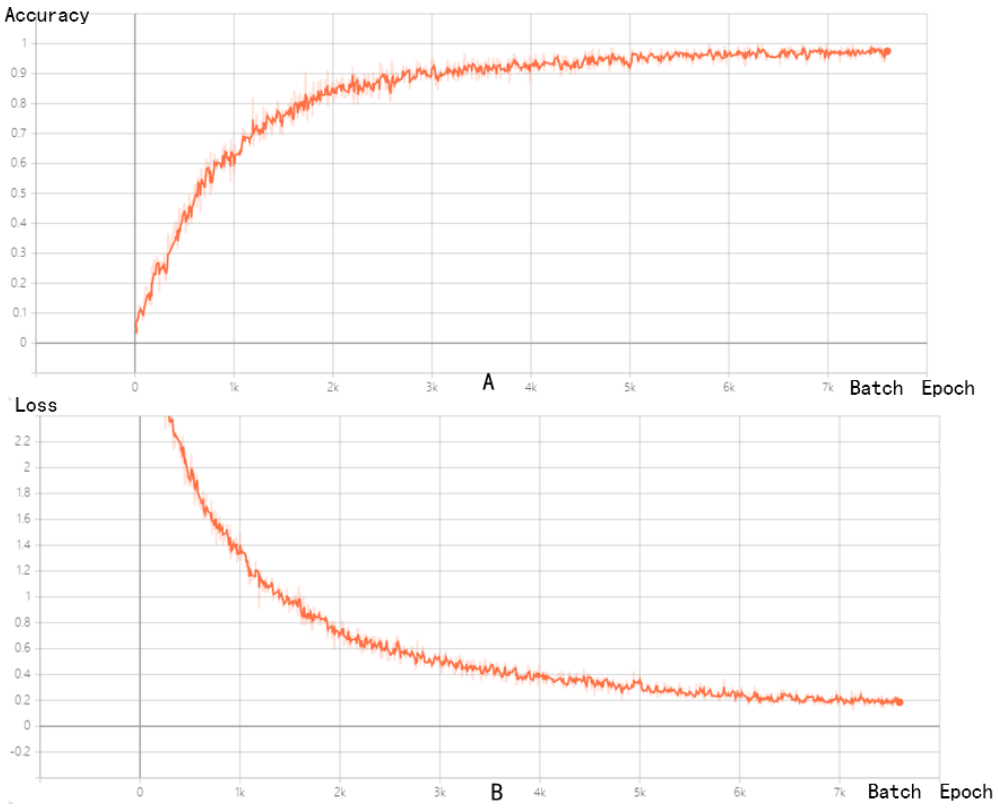

图 4.2.3-1A B 准确率与损失函数值变化图

实验结果表明深度神经网络具有较强的学习奶牛背部特征的能力，可以应用到复杂环境下的奶牛身份识别的任务中去。

4.3 VGG 方法

Alexnet 方法取得了不错的效果，我们想要在此基础上加深网络的深度和复杂度，进一步测试深度卷积神经网络对奶牛个体身份识别的效果，为此我们尝试了 VGG 网络。VGG 网络同样是非常经典的网络结构，其出色的特征提取能力，仍是当今许多工程项目的首选网络。与 Alexnet 相比，该网络有更小的卷积核、更小的池化层、更深的网络层次和更宽的特征图谱。实验表明，深度神经网络能很好的胜任奶牛个体分类的任务，加深网络的层数和节点的复杂程度有利于提升奶牛个体身份识别的准确率和鲁棒性。

4.3.1 模型搭建

选取 VGG 经典 19 层网络 VGG19 搭建网络模型。卷积核大小均为 3*3，池化层尺寸均为 2*2。前两层为 64 核的卷积层 conv1-1 和 conv1-2，随后接一个池化层 pool1；两个 128 核的卷积层 conv2-1 和 conv2-2 后接一个池化层 pool2；三个 256 核的卷积层 conv3-1、conv3-2 和 conv3-3 后接一个池化层 pool3；三个 512 核的卷积层 conv4-1、conv4-2 和 conv4-3 后接一个池化层 pool4；三个 512 核的卷积层 conv5-1、conv5-2 和 conv5-3 后接一个池化层 pool5。前面的结构相当于 Alexnet 的前五个卷积层（conv1-conv5），最后三层与 Alexnet 的结构相同，三个全连接层 fc6、fc7 和 fc8。其中 fc6 和 fc7 有随机失活 Dropout 操作，最后一个全连接层 fc8 后接 SoftMax 完成分类。下图（图 4.3.1-1）展示了详细结构：

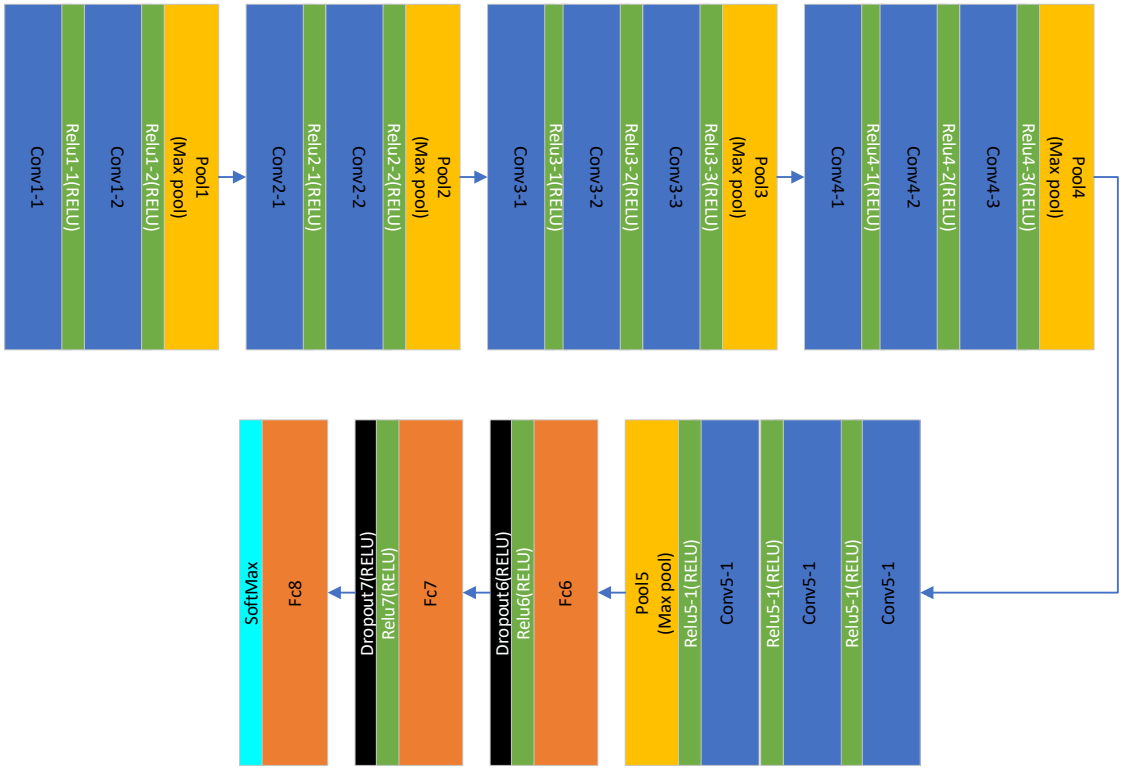

图 4.3.1-1 VGG19 网络架构图

4.3.2 训练方法

与 Alexnet 类似，我们采取迁移训练的方法对 VGG19 进行微调。VGG19 的网络层次更深，相比 Alexnet 只需要微调 fc8，VGG19 网络则需要对多个全连接层进行微调[8]，这里测试了 fc6、fc7 和 fc8 的微调情况。出现这个现象的原因是：网络层数越深，提取出来的特征就越复杂，需要对其进行分类的全连接层就越不容易分类，因此其全连接层的结构也就越复杂，迁移训练对全连接层的影响程度也越大。下图（图 4.3.2-2）展示了识别准确率随着迭代过程中上升情况和损失函数随着迭代过程的下降情况。可以看到在 1k~2k 次 batch epoch，即 15-30 次迭代过程中出现了挣扎的现象，陷入了

局部最优，但 VGG19 强大的特征提取能力很快就把模型的训练过程拉回到正轨上。随着迭代的进行，准确率直逼 99%。

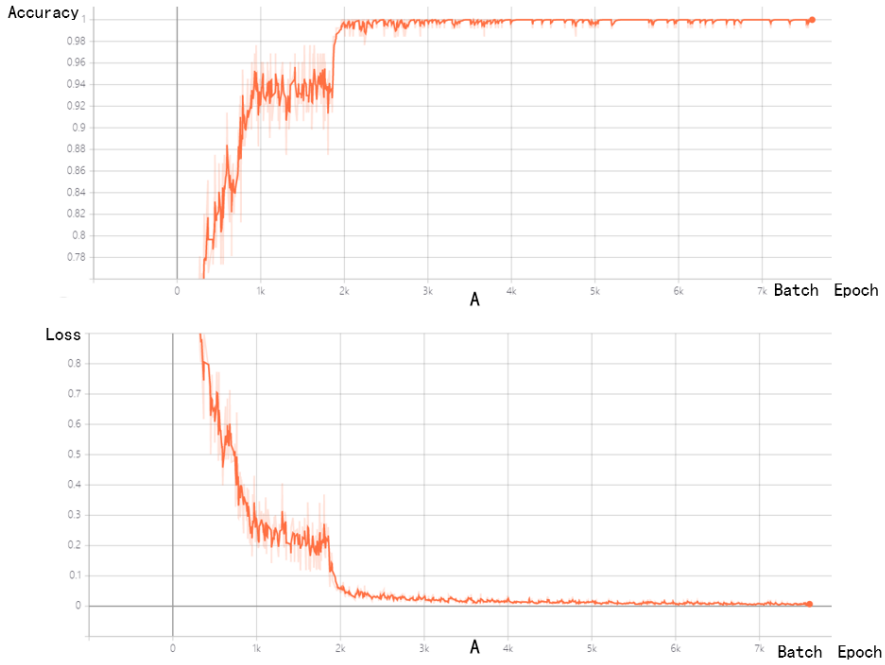

图 4.3.2-2 准确率与损失函数值变化图

4.3.3 结果分析

我们的奶牛侧身数据集（原始 202 张，共 15 类，扩增 52 倍，其中训练集 9453 张，测试集 1050 张。）在 Google Colab（GPU：Tesla K80 13GB）上以 0.001 的学习速率、128 的批处理大小下迭代 100 次，训练时长 9960.3 秒，测试单张图片时长约为 0.09 秒。下表（表 5.2.2-1）展示了模型在训练集、测试集和原始图片集上的准确率，其中训练集准确度为 100%、测试集准确率为 99.17%、平均准确率为 99.91%、原始数据集准确率达 100%。下表（表 4.3.3-1）展示了详细的检测准确率的情况：

表 4.3.3-1 VGG19 检测准确率

训练集	测试集	训练集+测试集	原始数据集
100.00%	99.17%	99.91%	100%

之所以取得较好的识别效果有两个原因：一是 VGG19 网络本身在提取图像特征具有优越性；二是采取了高效的迁移训练，而预训练的过程中本身就加入了奶牛图片，使预训练的网络本身就具有对奶牛图片的高度敏感性。美中不足的是原始数据集的图片本身较少，容易陷入过度拟合，这与数据采集不够充分有关。

实验表明，深度神经网络能很好的胜任奶牛个体分类的任务，加深网络的层数和节点的复杂程度有利于提升奶牛个体身份识别的准确率和鲁棒性。

4.4 对比

下表（表 4.4-1）展示了 SVM 方法、Alexnet 方法和 VGG19 方法的对比情况，不难发现深度神经网络在奶牛身份识别的任务上相比传统机器学习的方法更具有优势。其中相比 Alexnet 网络，VGG19 网络的识别效果要略优一些，这是因为 VGG19 比 Alexnet 具有更多的卷积层和池化层，有更强的特征提取能力，但是全连接层的数量和结构两者又是相等的。这也从侧面说明卷积神经网络的强大之处在于卷积层和池化层，而非全连接层。

表 4.4-1 结果对比

	SVM方法	Alexnet方法	VGG方法
准确率	71.43%	97.33%	99.19%
训练时间	3.86s	5881.68s	9960.40s
测试时间	0.0006s	0.02s	0.09s

以上是对奶牛个体身份识别的众多研究中最具代表性的三个方法。我们可以得到的基本结论是：SVM 等传统机器学习方法难以区分奶牛场复杂环境的前景和背景，导致识别准确率较低。而深度神经网络则具有较强的特征提取和识别的能力，即使是在背景复杂、奶牛侧身弯曲或不完整等情况仍具有较强的识别能力，和较强的准确性。同时，我们的研究表明，加深神经网络的深度和层数有利于体改识别准确率。要想进一步验证使用比 VGG19 更深的神经网络是否能继续提升识别的准确度和鲁棒性，需要采集更多类别的奶牛数据，同时每类奶牛的图片数量也要增加；但从整个神经网络的发展趋势上来看，结论是肯定的。

5 用户界面

上述内容完成后，为了更加直观的展示我们的项目成果，充分利用奶牛场的监控资源，我们开发了一款 B/S 架构的网站应用程序。选择使用 ASP.NET MVC 框架进行用户界面的开发，开发工具为 Microsoft Visual Studio。MVC 框架相较于 Web 窗体框架较为轻量。ASP.NET 主要使用 C#语言进行后台编程开发，操作比较简单。前台使用 Razor 引擎内嵌 C#代码，动态显示后台的奶牛图片、视频、训练过程等数据，并用 Ajax 方法与服务器端进行异步数据传输。利用 JavaScript 动态展示学习过程，搭载成熟的前端框架 Bootstrap，快速、便捷的美化界面。

该平台提供了奶牛场监控视频存放接口、奶牛个体图片分类接口、基于深度学习的奶牛个体身份识别模型框架。用户使用该平台可以方便的构建自己的奶牛身份数据库，通过监控视频来识别奶牛身份，达到对奶牛的实时监控和个体定位的目的。该平台实现了奶牛场监控视频资源的统一管理和奶牛个体身份的实时监控，对奶牛资源进行精细的加工利用，系统架构见下图（图 5-1）。

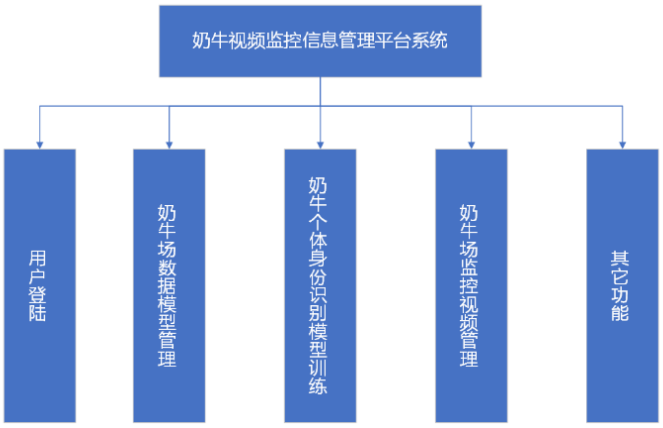

图 5-1 系统架构图

5.1 效果展示

首页为导航页面，页面结构如下图（图 5.1-1）所示。其中最上方为登陆页面。紧接着是导航栏，包括“首页”链接、“相关机构”下拉框、“主要功能”下拉框、“下载”下拉框和“关于”链接以及搜索框和搜索按钮。中间的主题部分是平台名称以及滚动播报栏，以及三个主要功能子系统的入口，子系统必须先登录后才可以进入，子系统与“主要功能”下拉框的内容相同。最下方显示网站的版权信息。

图 5.1-1 首页

登录界面采用了 modal 模态框弹出。由于使用了 MVF 框架，所以登录和演示并没有用 post 直接向后台传参数，而是用 JSP 编写了 AJAX 脚本进行更为灵活的参数传递。效果如下图所示（图 5.1-2）。由于本项目用户信息很简单，所以我们没有使用数据库，而是直接将用户的用户名密码保存到了文本文档里面。用户登录时将文本文档里面的值取出，与用户输入的信息进行对比，返回是否登录成功。

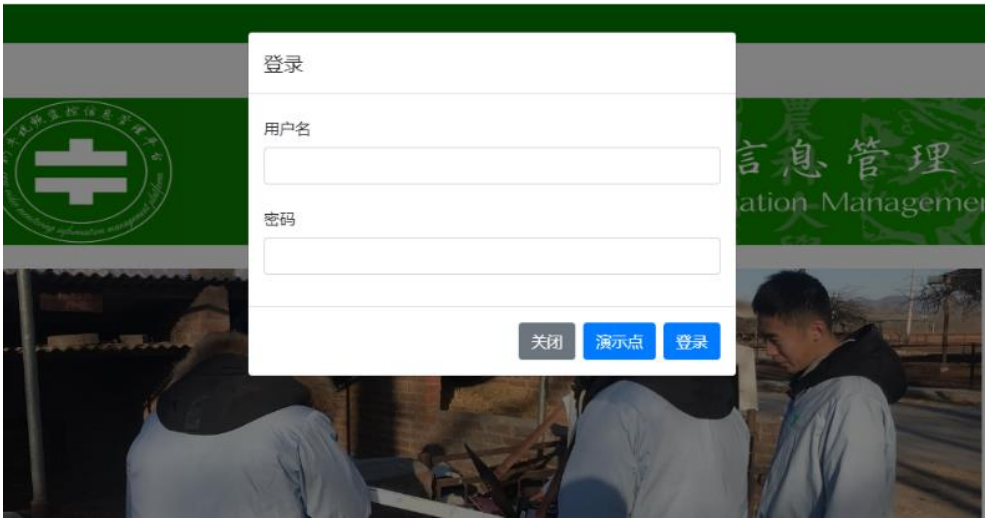

图 5.1-2 登录页面

用户登录成功后，可以进行数据库的编辑操作，如下图所示（图 5.1-3、图 5.1-4）。存放奶牛图片的目录下，有多个子目录，每个子目录是一个奶牛场或区域的集合。在打开网页后会动态的出现在上方选择栏目里面。子目录下是 1 个区域介绍和多组奶牛图片。区域介绍的文件名为 introduction.txt，里面的内容会被现实到网页上。多组奶牛图片中，每一组都是一头奶牛的多张图片，每组取出第一张图片作为该组的封面。奶牛图片均采用.jpg 的图片格式。可以在网页中编辑分组后动态添加删除或下载。

图 5.1-3 编辑牛棚

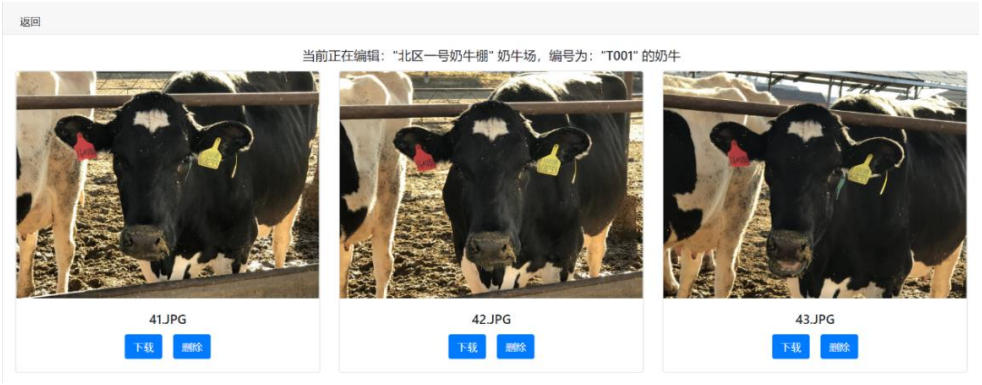

图 5.1-4 编辑奶牛

同时用户还可以对奶牛的监控视频进行观看，如下图所示（图 5.1-5）。视频的存储格式与图片类似。用于存放奶牛视频的目录下，有多个子目录，每个子目录是一个奶牛场或区域的集合，在打开网页后会动态的出现左侧的牛场选择中。每个子目录下又有多个子目录，用于表示不同的摄像头或视频，在打开网页后会出现在上一级目录的下面。每个视频子目录下只有一个名为 video.mp4 的视频会被动态显示在网页中。

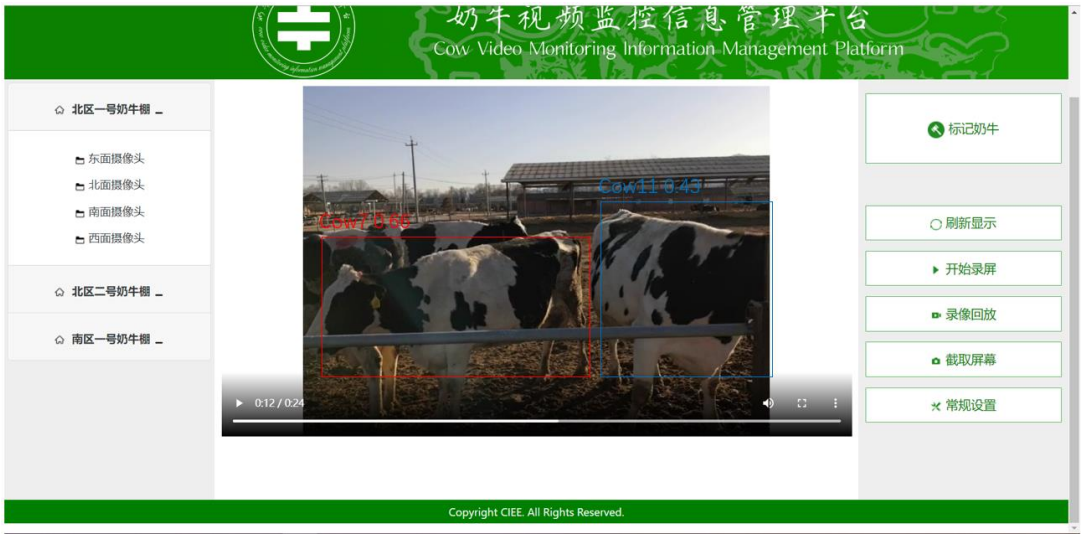

图 5.1-5 查看监控视频

用户在登陆后可以通过点击首页下方的“深度学习模型训练”按钮或点击导航栏的“主要功能”下拉框的“深度学习模型训练”链接进入该子系统，该子系统的界面如下图所示（图 5.1-6，图 5.1-7）。在奶牛数据集编辑完毕后，可以通过该子系统对奶牛个体身份识别模型进行训练。通过该子系统可以对不同牛场的奶牛数据集进行相应的训练，查看每个奶牛场的训练精度，开始训练，暂停训练和继续训练。同时可以查看训练日志和清除日志。训练日志中记录了有关日志的位置，图片的位置和模型的位置，以及开始时间、结束时间和暂停时间。还包括了训练过程中详细的准确率和损失函数的有关情况。

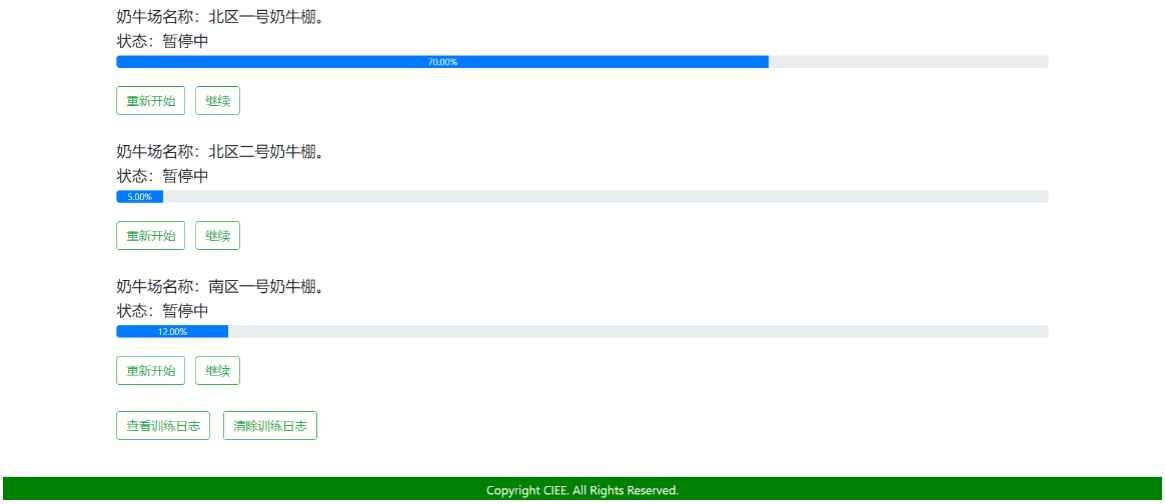

图 5.1-6 训练情况图

```
tarinlogloc: C:\Users\lenovo\Desktop\CowCow\CowCow\Python\trainlog.txt
picloc: C:\Users\lenovo\Desktop\CowCow\CowCow\CowPicture\三元养牛场
resultloc: C:\Users\lenovo\Desktop\CowCow\CowCow\Python\result\三元养牛场.txt
netsaveloc: C:\Users\lenovo\Desktop\CowCow\CowCow\Python\save\三元养牛场
maploc: C:\Users\lenovo\Desktop\CowCow\CowCow\Python\map\三元养牛场\三元养牛场-
traintimes: 100
nowtimes: 0
newtrain: true
训练开始，时间：2020-01-22 17:21:11.547520
0/100: 损失函数：1.7315946; 当前准确率：10.9375
1/100: 损失函数：0.9090013; 当前准确率：56.25
2/100: 损失函数：0.4503166; 当前准确率：89.0625
3/100: 损失函数：0.19017959; 当前准确率：96.875
训练暂停
结束时间：2020-01-22 17:22:38.703572
```

图 5.1-7 训练日志图

5.2 关键步骤实现方法

本平台难点在于实现后端 C#语言与深度学习框架 TensorFlow 的有机结合，协作完成训练工作，并动态显示到网页上。如何解决这一问题，成为完成该平台的关键。

首先，对于 python 的代码部分，使用 PyInstaller 进行打包，将其转化为可执行文件，这样可以避免 python 环境和路径问题对平台使用的影响。对于打包好的 exe 文件，我们使用命令行传递参数的形式，将需要操作的配置文件的地址传入到文件中。配置文件采用了字典的数据结构，包含 10 个参数，以及其对应的功能见下表（表 5.2-1）。浏览器每个 2s 会发出一次 AJAX 请求，服务器会将训练信息的记录返回给浏览器，浏览器再进行动态的信息显示。

表 5.2-1 python 网络训练配置参数

编号	名称	功能
1	traintimes	网络训练的上限次数
2	nowtimes	当前网络的训练次数
3	stop	是否停止训练进程
4	newtrain	是否为重新训练
5	piclocation	图片的路径
6	resultloc	训练的信息记录的文件
7	saveloc	模型的保存路径
8	tarinlogloc	训练日志的路径
9	maploc	编号的映射关系的路径
10	errorlogloc	错误日志的路径

在视频检测的过程中，在点击标记奶牛按钮后，服务器将会打开上文中打包好的检测程序。该程序包含了一个文件监听器，用于监听视频检测配置文件的信息变化。在配置发生修改的时候，会自动读取配置文件里面内容，并且开启一个新的线程去执行。服务器后端，每隔 0.5s 修改一次配置文件，修改的同时会被检测程序监听到，并进行相应的识别操作。识别结束后，检测程序通过 OpenCV

创建一个 alpha 通道为 0 的透明图像，并在图像中画好加测到的方框，标记检测到的类别，最后将该图像保存。由于服务器没有办法主动联系前端并发送检测完成的消息，所以浏览器每隔 0.5s 会刷新一次图片地址，并将获取到的新的标记好的图像叠加到视频上面。

在服务器后端打开监听检测进程的时候，同样需要传入配置文件的路径。配置文件中包含了一下几个参数（表 5.2-2），并由字典的数据结构保存。

表 5.2-2 视频监测配置参数

编号	名称	功能
1	netloc	保存网络模型的路径
2	videoloc	当前播放视频的路径
3	videotimes	当前视频播放时间
4	ffmpegloc	ffmpegloc的位置
5	map	编号的映射关系的路径
6	resultpicloc	识别结果的图片路径
7	temppicloc	存储临时截图的路径
8	stopprocess	是否停止进程的标志

6 结论

在本项目中，我们先后进行了奶牛身份数据集的采集与制作，奶牛目标检测，奶牛个体身份识别以及奶牛视频监控信息管理平台的搭建。我们致力于将深度学习的手段应用到奶牛养殖管理这一领域中去。其中奶牛目标检测结果显示深度神经网络具有在复杂环境下对奶牛个体进行目标检测的能力，可以应用到实际的奶牛场养殖中检测奶牛个体的任务。而奶牛的身份识别结果显示，深度神经网络具有在复杂环境下对奶牛个体进行身份识别的能力，可以应用到实际的奶牛场中识别奶牛身份识别的任务。我们将上述研究成果整合至了奶牛视频监控信息管理平台中，用户使用该平台可以方便的构建自己的奶牛身份数据库，通过监控视频来识别奶牛身份，达到对奶牛的实时监控和个体定位的目的。实现了奶牛场监控视频资源的统一管理和奶牛个体身份的实时监控，对奶牛视频资源进行精细的加工利用。

7 参考文献

[1] 吕伟国. 基于 EPC 物联网和 RFID 的奶牛精细养殖信息管理系统[D].吉林大学,2013

[2] 刘健,袁谦,吴广,喻晓.卷积神经网络综述[J].计算机时代,2018(11):19-23.

[3] Xia M, Cai C. Cattle face recognition using sparse representation classifier[J]. ICIC Express Letters, Part B: Applications, 2012, 3(6): 1499－1505.

[4] 赵凯旋,何东健.基于卷积神经网络的奶牛个体身份识别方法[J].农业工程学报,2015,31(05):181-187.

[5] 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.

[6] 刘杰鑫,姜波,何东健,宋怀波.基于高斯混合模型与 CNN 的奶牛个体识别方法研究[J].计算机应用与软件,2018,35(10):159-164.

[7] Krizhevsky Alex, Sutskever I., Hinton G. ImageNet Classification with Deep Convolutional Neural

Networks[J]. Advances in neural information processing systems, 2012,25(2).

[8] Girshick R , Donahue J , Darrell T , et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C] CVPR. IEEE, 2014.

[9] W. Andrew, C. Greatwood, T. Burghardt. Visual Localisation and Individual Identification of Holstein Friesian Cattle via Deep Learning: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), 2017[C].2017 22-29 Oct. 2017.

[10] Simonyan K , Zisserman A . Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.

[11] Shorten C , Khoshgoftaar T M . A survey on Image Data Augmentation for Deep Learning[J]. Journal of Big Data, 2019, 6(1):60.

[12] Ren S , He K , Girshick R , et al. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2015, 39(6):1137-1149.

[13] Joseph Redmon, Santosh Divvala , Ross Girshick, Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection[J]. Computer Science, May 2016.