成员分工

杨再俨：完成整体代码的书写，数据集的收集与整理，模型的训练与测试，产品介绍以及实验、算法设计等部分的文档

陈昊华：完成CNN&Mobilenet模型的学习与相关文档的编写，提出创新方法并完成实验总结

**一、产品名称**

兽友-动物识别系统

**二、产品介绍**

兽友-动物识别系统是一款基于CNN的mobilenet模型设计的图像识别系统，能够识别生活中常见的动物并给出简单的介绍。此产品适用于各个年龄段的对大自然动物界感兴趣的人群，有助于培养对自然界的认知，促进人与自然和谐相处，有益于人与自然和谐共生。

产品上手容易，使用简单，界面简洁。用户仅仅只需上传图片并点击识别即可得到识别信息，本产品针对年幼的小朋友学习课外知识是一个不错的选择。

本产品的设计灵感源于《机器智能》中的一句话——“根据科学技术发生和发展的辅人律和拟人律，机器智能研究的唯一价值就在于更好地辅助人类拓展自己‘认识世界和改造（优化）世界’的能力。”作为一名喜爱观看纪录片并探索大自然的爱好者，往往在学习或生活中会遇见一些不知名的小动物。苦于当今市面上并没有一款简洁小巧的专业动物识别系统，因此常会错失许多认识小动物的机会。得益于这段时间以来对人工智能领域的学习，让我接触到了模拟生物神经网络的BP神经网络、能够在生成图像以及识别图像的对抗过程中得到人眼无法分辨图像的GAN生成式对抗网络等，我具备了基本的借助这些“机器智能”帮助解决生活中难题的能力。因此，经历了收集数据集、选择模型、训练并调整参数等过程，最终构建了这款动物识别系统。

**三、实验设计**

**1.CNN&mobilenet模型概述**

卷积神经网络在沉睡了近20年后,如今成为了深度学习方向最主要的网络结构之一.从一开始的只有五层结构的LeNet, 到后来拥有19层结构的VGG, 再到首次跨越100层网络的Highway Networks与ResNet, 网络层数的加深成为CNN发展的主要方向之一。随着CNN网络层数的不断增加,gradient vanishing和model degradation问题出现在了人们面前,BatchNormalization的广泛使用在一定程度上缓解了gradient vanishing的问题,而ResNet和Highway Networks通过构造恒等映射设置旁路,进一步减少了gradient vanishing和model degradation的产生.Fractal Nets通过将不同深度的网络并行化,在获得了深度的同时保证了梯度的传播,随机深度网络通过对网络中一些层进行失活,既证明了ResNet深度的冗余性,又缓解了上述问题的产生. 虽然这些不同的网络框架通过不同的实现加深的网络层数,但是他们都包含了相同的核心思想,既将feature map进行跨网络层的连接。

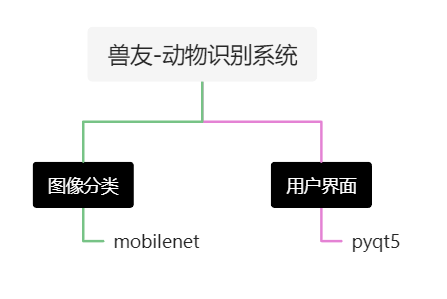
MobileNet的基本单元是深度级可分离卷积（depthwise separable convolution），其实这种结构之前已经被使用在Inception模型中。深度级可分离卷积其实是一种可分解卷积操作（factorized convolutions）

**2.架构设计**

产品核心功能是能够自动处理并分析输入的图像，给图像分类，也就是图像识别任务。mobilenet是一款基于CNN的轻量级网络，所以要熟悉mobilenet我们需要了解如何使用卷积神经网络进行图像分类任务。

除了核心的模型部分外，一个明朗、简洁的用户界面也必不可少。本产品的用户图形化界面使用可用于python2和python3的GUI库pyqt5进行开发。

产品架构如下：



**3.算法设计**

产品的算法设计分为两部分，一部分用于训练模型，另一部分用于构建图形化界面。

**3.1 图形化界面开发**

用户使用系统时会直接弹出用户界面等待用户等待用户鼠标操作，当用户需要关闭系统时点击右上角的关闭按钮并确认关闭即可。

用户界面设计流程为：实例化QApplication对象对系统进行管理——>实例化MainWindow对象并调用show方法展示界面——>调用exit方法退出系统。

MainWindow类中定义有如下函数：

def \_\_init\_\_(self)

#初始化系统，包括创建窗口、初始化logo、系统名称、模型加载、初始化用户界面等

def initUI(self)

#初始化用户界面，基本设计思想就是设计各种布局组件并将组件按照想要的规律进行排版，

#组件中包含动物名称窗口、动物简介窗口、“上传图片”按钮、“开始识别”按钮等

def change\_img(self)

#这个函数用于处理用户点击“上传图片”按钮后触发的事件，

#函数会打开文件资源管理器等待用户选择图片，用户选择图片后系统对图片进行处理并在界面展示给用户，

#同时系统会调整图片大小便于后续图像识别

def predict\_img(self)

#该函数用于处理用户点击“开始识别”按钮后触发的事件；

#函数会将change\_img()处理过的图片读取并转换为numpy数组，将数组作为参数传入识别模型中识别，

#识别完成后使用result接收识别结果，通过判断识别对象的名称给出相应的介绍

def closeEvent(self, event)

#该函数用户处理用户点击右上角关闭按钮触发的事件，防止用户误操作而关闭页面，

#具体实现为使用reply记录用户是否确认退出，然后进行if else判断选择是否关闭页面

**3.2 模型训练**

训练模型之前应该做如下准备工作：导入数据集、导入模型、指定训练轮数。关于数据集的设计和收集会在下一节介绍，系统借助tensorflow2.0实现MobileNetV2轻量化网络结构。准备工作完成后开始训练，训练完成后记录训练data并保存训练后得到的模型，绘制模型训练过程图。

本部分使用到的函数如下：

def data\_load(data\_dir, test\_data\_dir, img\_height, img\_width, batch\_size)

#数据集加载函数，加载用于训练的数据集（测试集、验证集）并设置训练的batch大小，参数含义如下：

# @data\_dir 训练集目录

# @test\_data\_dir 测试/验证集目录

# @img\_height,img\_width 归一化处理的height以及width

# @batch\_size 图片一批一批的加载，n张一批

def model\_load(IMG\_SHAPE=(224, 224, 3), class\_num=12)

#借助tensorflow加载预训练的mobilenet模型，此处选择优于MobileNetV1的MobileNetV2

#加载完成后对模型进行必要的设置如冻结主干参数、归一化处理、选择optimization等

def show\_loss\_acc(history)

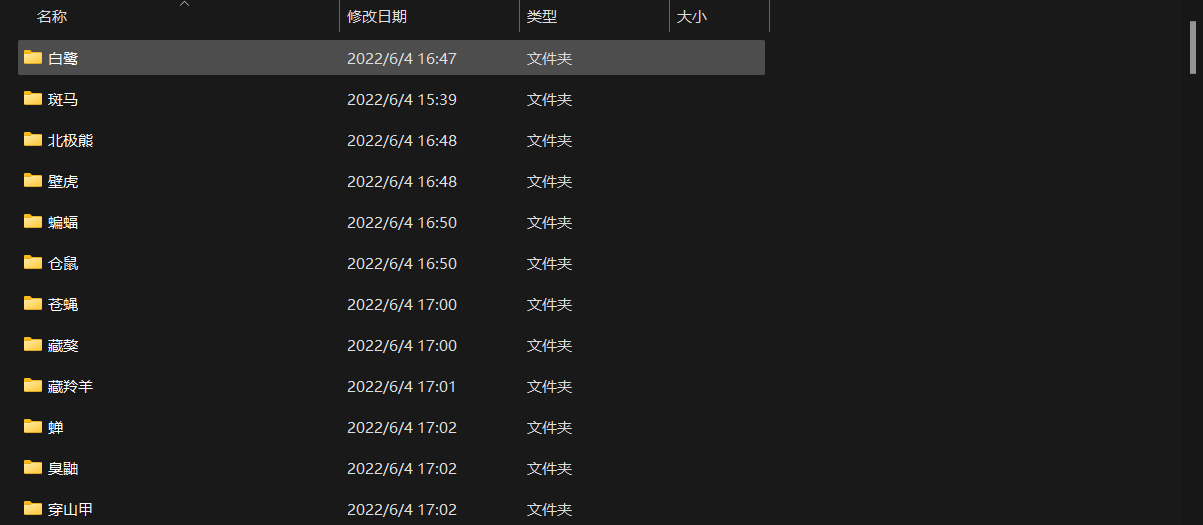
#该函数用于展示模型在训练过程中的accuracy以及loss变化，相比于直接看training data更加直观

def train(epochs)

#该函数是具体的模型训练函数，训练指定的epoch后得到模型并保存在当前目录的models目录下

**4.数据集设计**

为了尝试使用自己收集的数据集进行训练，这里使用的是个人收集的数据集，未来假如要扩充系统识别动物的种类可使用个人收集+网络数据集的方法。通过对收集得到的图片以动物名称进行分类，删除无用图片以及未加载图片等预处理操作后得到可使用的数据集如下：



通常数据集应分为训练集、验证集和测试集，由于此处数据集数量并不是很大，所以并未划分测试集，在后续对模型的测试过程中使用验证集充当测试集。



**四、实验结果**

**1.碰到的问题&解决方法**

### 1.1 环境配置

首先搭建环境的过程中遇见了许多问题，如权限不够、环境变量缺失、包管理器版本过低等，所以花费了一定时间进行环境的配置。我们利用了视频教学以及利用Google查询资料成功的解决了开发所需环境的配置问题。

### 1.2 数据集的收集与整理

因为并没有在网络上找到合适的数据集，所以我们使用了爬虫的方法自行收集数据集，但是在收集数据集的过程中出现了一些问题。首先是爬虫根据关键字进行索引，所以当爬取某些动物名称时会得到大量动漫图片，这对于我们需要识别自然界中的动物会产生很大的干扰；同时因为网络或其他原因，有些图片存在损坏和缺失的问题，故会导致最终模型在训练过过程中崩溃而终止。解决方法是通过人工整理并对图片进行分类，最后得到完整、有效的动物数据集。

### 1.3 模型的训练

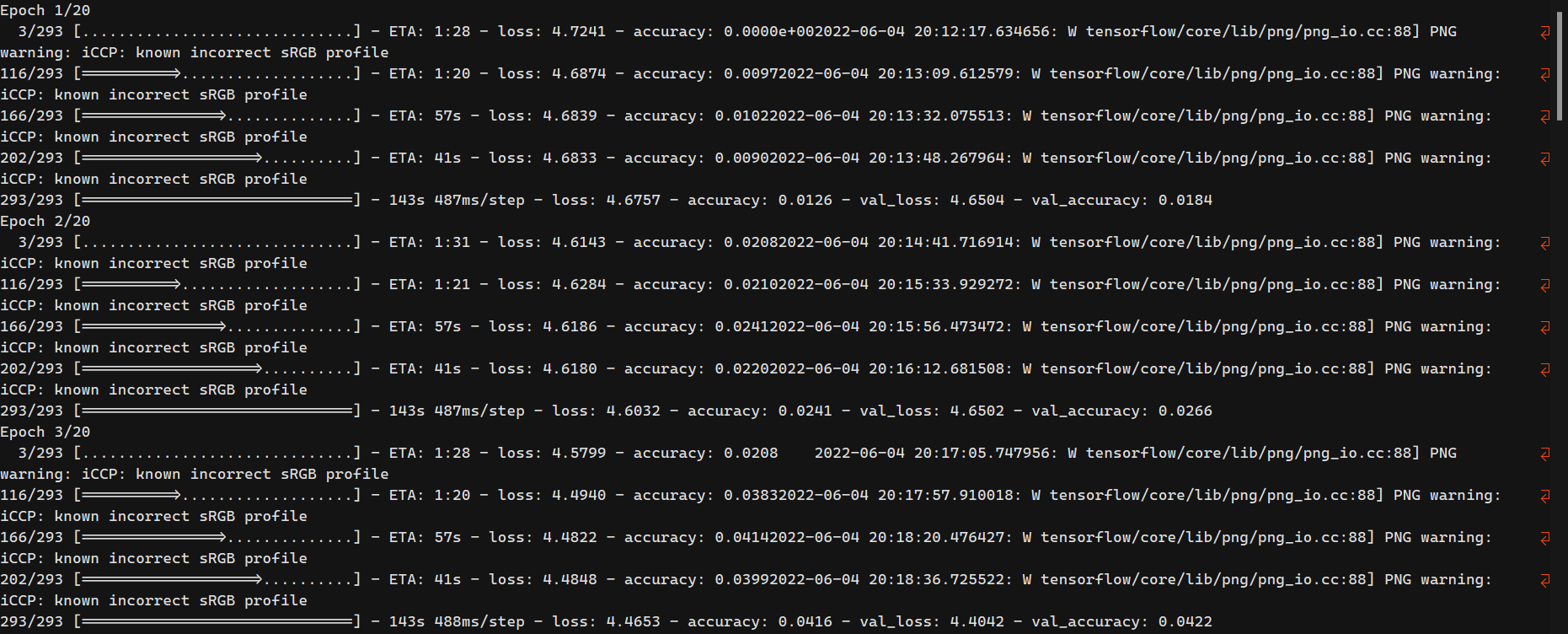
模型训练过程中的参数设置也非常重要，在训练CNN模型的过程中因为参数设置不当且epoch设置的过大，导致花费了非常长的时间来训练而最后得到的训练模型不论是在测试过程还是实际使用的过程中都出现了非常差的效果。我们根据一定时间的摸索最终确定了最佳的训练参数。

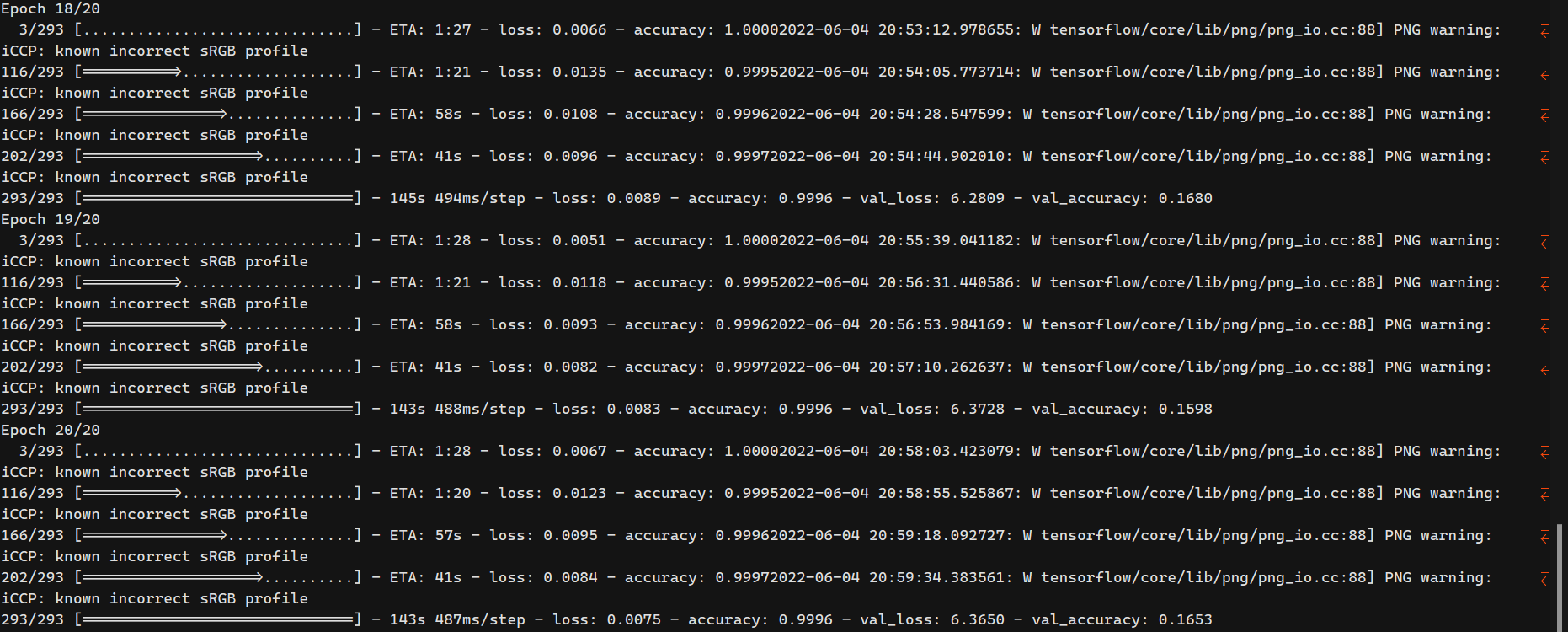
**2.创新方法**

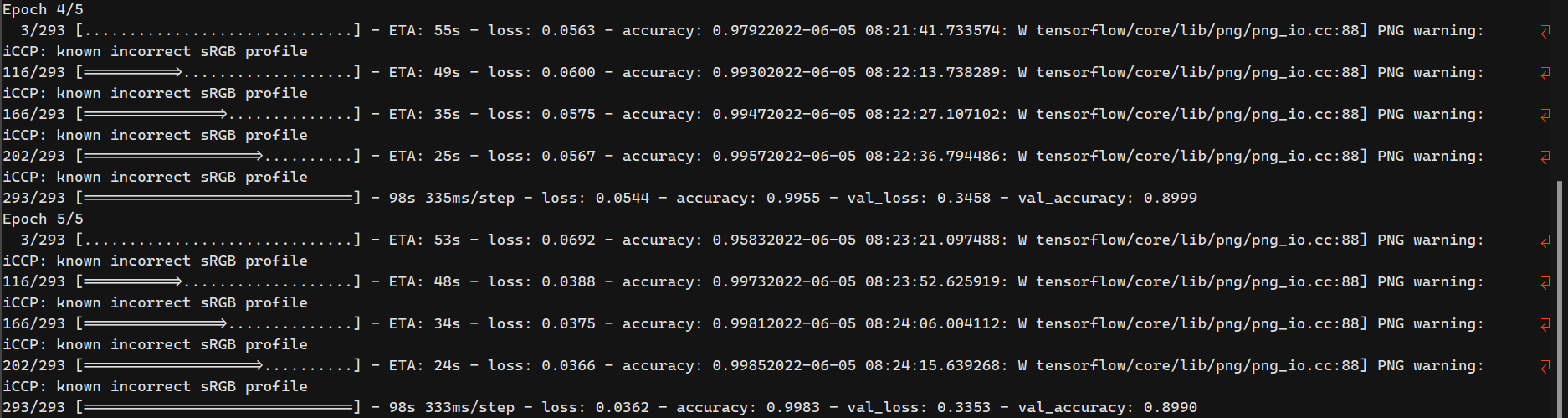
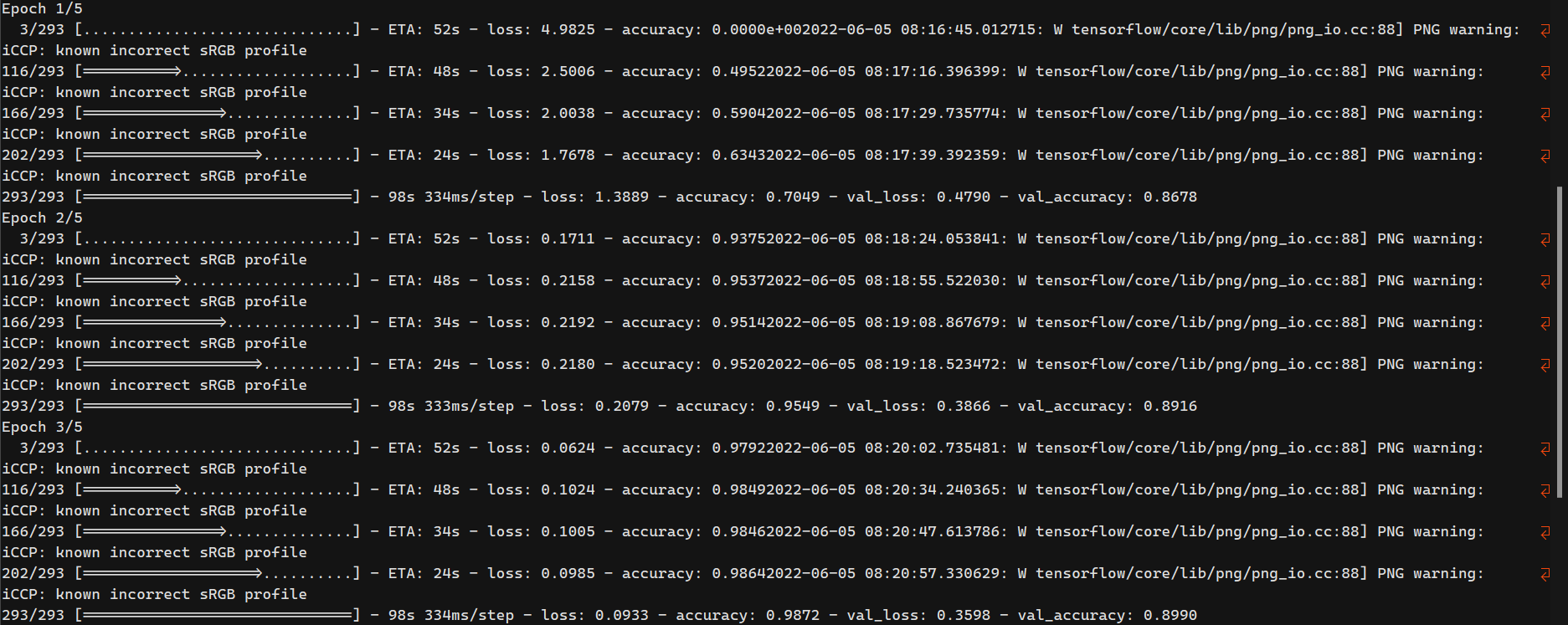
我们这个动物识别系统在市面上还没有，即使存在的体积也较大，不方便安装和使用，所以我们这个动物识别系统可以最好程度的大众化。在某些真实应用场景中，相比于传统的CNN算法，我们这个模型可以更好的对动物进行识别。同时我们的算法在参数调整上更加灵活一些，相比较传统CNN的参数调整更加敏感。我们的算法还用了爬虫来爬取我们需要的样本，这使得样本选取更加灵活。

**3.结果分析**

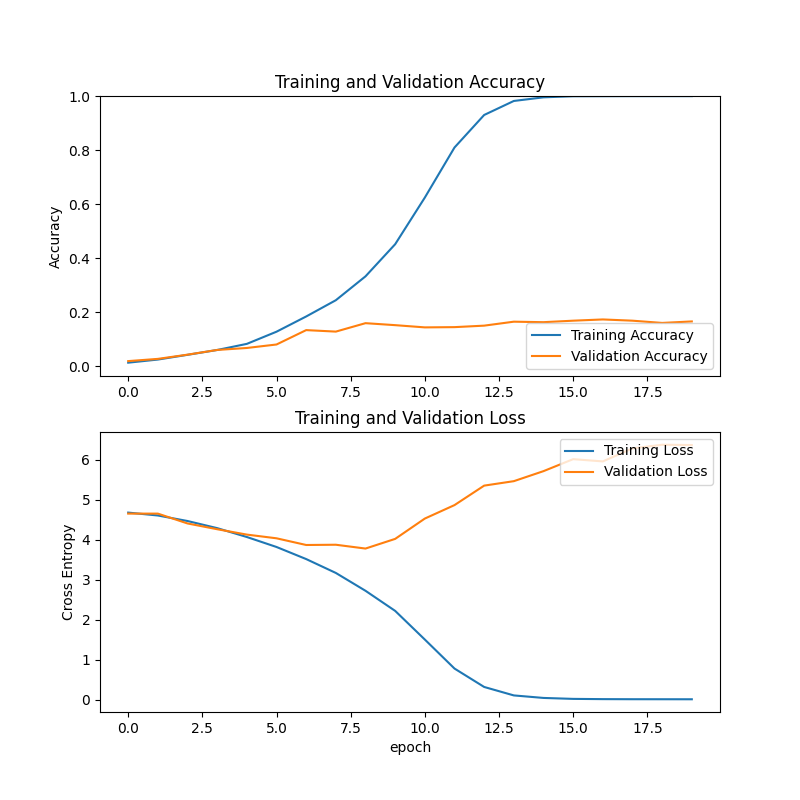
最初是打算选择CNN作为图像识别的模型，但是经过训练和测试以及实际使用结果来看，结果并不理想。如下分别是CNN和mobilenet在训练过程中的部分training\_data以及模型训练过程图：



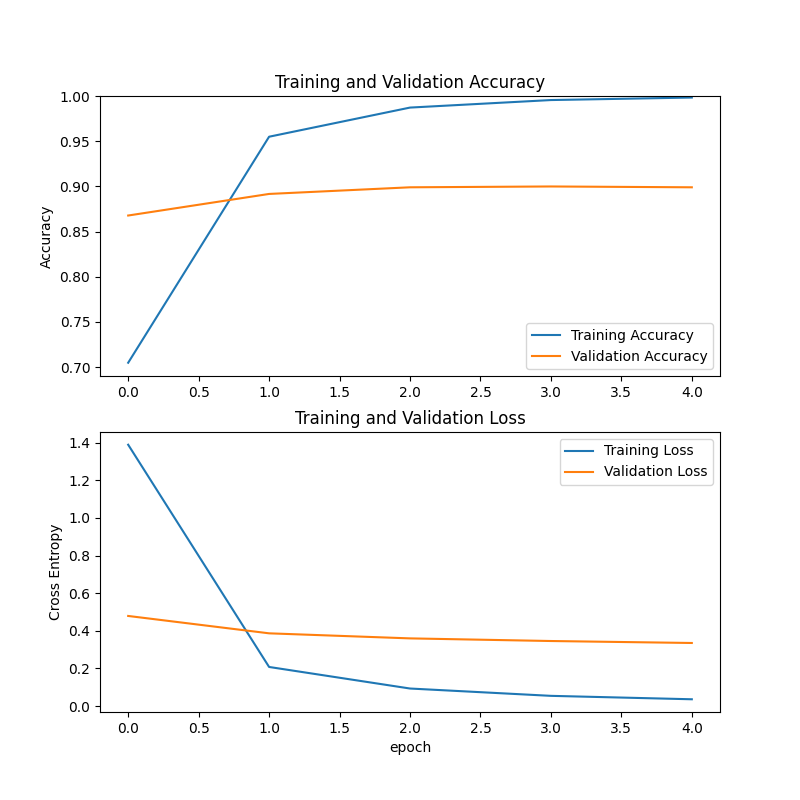




可以看到尽管CNN训练轮数多达20个epoch，但是它的准确率仅仅只在训练集上提升，而在测试集上准确率始终维持着较低的水平，甚至在后来的训练过程中由于过拟合的产生导致准确率在验证集上下降：



而mobilenet模型仅仅只训练了5个epoch就能够在训练集以及测试集上得到漂亮的结果，同时将得到的模型用于实际的系统中识别动物也可以得到正确的结果：



最后展示使用训练得到的CNN模型和mobilenet模型对互联网上的某张图片进行识别的效果：



**4.讨论与思考**

通过该实验我们能够掌握如何使用pyqt5进行简单的图形化界面设计，并且巩固对爬虫的运用，且通过实验过程中对参数调节以及实验过程中遇到的问题有了更具体深刻认识。对于CNN的在生活中的运用也给了我们很多启发。最后我们通过本次实验可以看出除了算法和参数外，训练样本对于识别能力的影响是极其显著的。关于该实验实际上我们还可以进一步深化研发出相应的图像识别系统，最终整合形成一个巨大的数据库，实现万物可识别。