猫狗识别

杨安东

目录

摘要

本实验的目的是在 kaggle 数据集上进行猫狗识别。在查阅资料后最初决定使用 ResNet 网络来实现猫狗识别，由于使用笔记本进行实验，首先尝试较为简单的 ResNet18，运行速度过慢，无法获得理想结果。最后改用 AlexNet 实现猫狗识别，最终获得了本报告的结果。

1 问题描述

本实验的目标是使用 kaggle 的猫狗数据集进行训练，之后使用训练好的模型识别猫狗图片。由于 kaggle 数据集中只有训练集有标签，测试集没有图像标签，因此需要人工从训练集中划分测试集与训练集。

2 解决方案

在图像分类方面，使用较广的网络有 VGGNet、GoogLeNet、ResNet、AlexNet 等，考虑到此问题是二分类问题，不需要过于复杂的网络。因此考虑使用 ResNet18 进行实验。

为了获得较高的准确率，输入图像计划使用较大的 224*224 大小，同时对不同通道进行不同的标准化，对于红黄蓝三个通道标准化的期望 (mean) 与标准差 (std) 分别为 [0.485, 0.456, 0.406] 和 [0.229, 0.224, 0.225]。标准化使用的公式为：

$$output[channel] = (input[channel] - mean[channel])/std[channel]$$

在超参调节方面计划使用微软的 AutoML 工具 nni 进行超参调节。

2.1 网络结构设计

最初计划使用 ResNet18 进行实验，ResNet 引入了残差网络的概念。如图 1所示，一个残差块将其输入按照一定权重与残差块的输出结合，可以有效避免梯度消失，进而加深了可以实现的网络深度，增强了网络的表达能力。

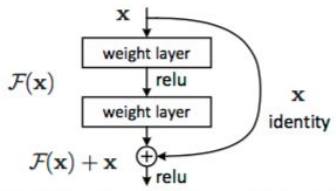

Figure 2. Residual learning: a building block.

图1 ResNet 残差块结构

将残差块依次链接，即可获得一个很深的网络，而残差块数量的不同形成了不同的 ResNet 网络，如图 2所示。由于本地算力不多，因此使用较浅的 ResNet18 进行实验。

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2 $\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

图2 ResNet 网络类型

但在之后的复现中发现低估了 ResNet18 需要的算力，实验在本地的 mac 上运行，在运行了 1 小时之后尚未完成训练进度的一半，考虑之后还可能需要调参优化，耗时过多。因此改用 AlexNet 进行实验，神经网络结构如图 3所示。可见相对于 ResNet 其没有残差块结构，深度也浅了很多，需要的算力更少。

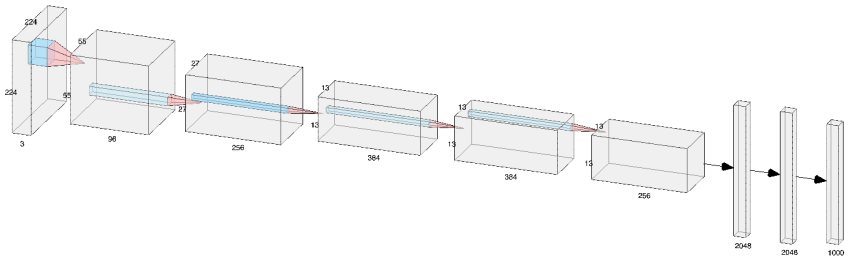

图3 AlexNet 网络结构

2.2 损失函数设计

损失函数的选择没有太多考虑，使用图像分类较为常用的交叉熵损失函数。

2.3 超参设计

计划使用微软的 AutoML 工具 nni 进行自动调参，但在了解后发现 nni 需要进行反复的训练寻找最佳参数，较为耗时，且需要指定，寻找范围与步数，相对于本实验而言并没有直接手动调节高效率，因此放弃使用 nni，改为直接手动调节。在综合考虑耗时与准确率并观察实验结果后，最终使用的超参如下：

1. lr=0.0001
2. batch_size=32
3. epoch=10

此外实验过程中发现学习率会显著影响训练效果，在本实验中尝试了 0.01，0.001，0.0001，0.00005 四个学习率，只有 0.0001 可以有效收敛。

2.4 创新点

1. AlexNet 输入数据处理，不同通道使用了不同的标准化参数。
2. nni 工具使用

3 实验分析

3.1 数据集介绍

数据集使用 kaggle 的猫狗数据集，数据集共 25000 张照片，其中训练集猫狗照片各 12500 张，这些图像具有标签，而验证集图像不具有标签。因此将训练集图像进行切割，使用 10000 张猫图像与 10000 张狗图像共 20000 张图像进行训练，使用 2500 张猫图像与 2500 张狗图像，共 5000 张图像进行测试。由于图像的大小不是统一的，因此需要对图像大小进行处理。为了保留更多的信息，每张图片处理为 224*224 大小，之后送入网络进行训练。

3.2 实验结果与分析

训练过程中损失函数变化如图 4所示。

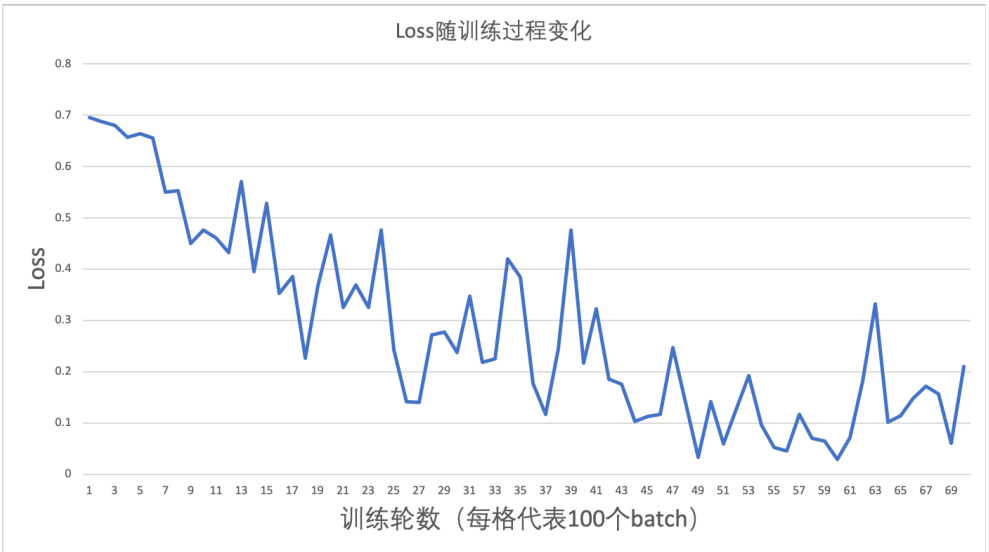

图4 Loss 随训练过程变化

训练进行 10 轮，获得了 89% 的平均准确率。最高准确率可以达到 90% 以上，如图 5所示。

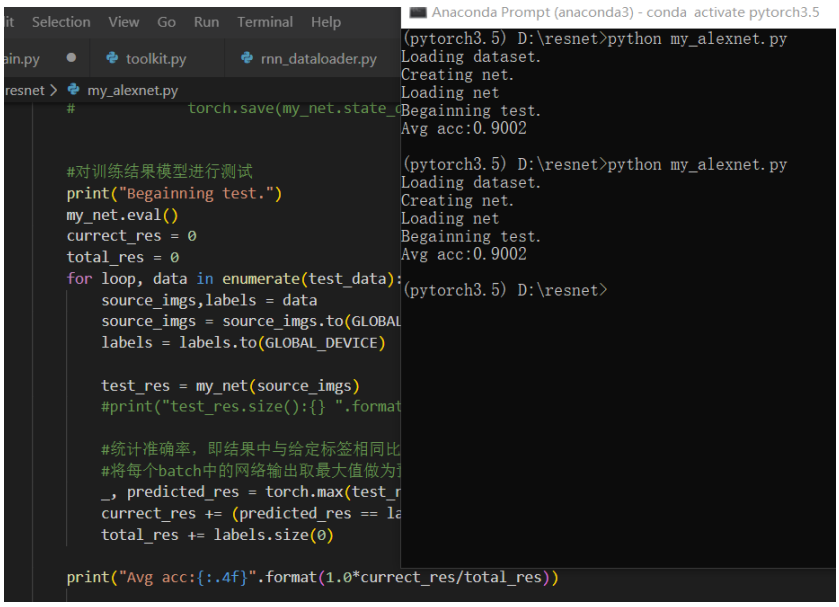

图5 测试集准确率

4 总结

在方案设计初期由于对各个网络需要的算力了解不足，使用 ResNet18 进行试验，结果因为训练过慢而无法获得结果。同时计划使用微软的 AutoML 工具也因为需要反复训练需要大量时间而被放弃，最终手动调节超参获得了尚可结果。