经典网络模型 ResNet-50 在 ImageNet-1k 上的研究 | 实验笔记+论文解读

原创

XianxinMao 于 2021-09-17 09:50:54 发布 3034 收藏 27

分类专栏： 神经网络与深度学习 文章标签： 深度学习 pytorch 机器学习

本文链接：https://blog.csdn.net/XianxinMao/article/details/120342693

需要 ImageNet-1k 数据集的来这篇博文: https://blog.csdn.net/qq_39377134/article/details/103128970

但是要准备好 240 GB 大小的磁盘空间哈，因为数据集压缩包是 120 GB 多一些。

本文是关于 ResNet-50 在 ImageNet 上的实验研究，目前的话，实验数据集分别是 ImageNet-240 和 ImageNet-1k，其中前者是后者的一个子集。

接下来直接上实验结果吧，第一次实验，我是 freeze all layer exclude last layer，注意此处我加载了在 ImageNet-1k 上预训练的模型，实验结果是: train_acc = 93.8, val_acc = 93.44, test_acc = 93.48。

第二次实验(这一次实验加载的预训练模型是加载第一次保存下来的模型，原因是我们要在第一次训练完 last layer参数的情况下做微调，而不是直接 unfreeze layer4 和 fc layer，这样会破坏预训练学习到的信息，这里还要提一下，微调的话，学习率最好是第一次训练的 1/10)，我是 freeze all layer exclude layer4 and fc layer(也就是上面说的 last layer)，实验结果是: train_acc = 95.24，val_acc = 93.6，test_acc = 93.7。对比第一次和第二次的实验结果，我们可以发现在 val 和 test 上获得了些许提升，但是从 train 可以看出开始过拟合了。

第三次实验(这一次实验加载的预训练模型是加载第一次保存下来的模型)，我是 freeze all layer exclude layer3 and layer4 and fc layer，实验结果是: train_acc = 95.81, val_acc = 93.67, test_acc = 93.9，对比第三次实验和第一次实验，可以发现，unfreeze 更多的网络层数，能略微提升准确率，但是也不太多吧。

对于过拟合，我说说我的看法吧，加大数据量可以缓解过拟合，但也仅仅只是缓解，除非你的数据集包含了所有现实情况，不然这个无法避免，我们能做的只是缩小 train_acc 和 val_acc 之间的 gap。

最后再说一下在 ImageNet-1k 上的 acc = 87.43，对比 ImageNet-240 和 ImageNet-1k 上的结果，我们可以发现，模型在 ImageNet-1k 上做 pre-train，然后 transfer 到 ImageNet-240 上，可以明显提升模型效果，不过两者都是同属于一个 domain，我们需要更多的在不同 domain 上测试 transfer 的效果才行。

对于数据量和模型泛化性的研究，有一篇论文写的很好，Big Transfer (BiT): General Visual Representation Learning，该论文在 ImageNet-1k(1.28M 张图片)，ImageNet-21k(14.2M 张图片)，JFT-300M(300M 张图片)，上分别实验，发现数据量越大，效果越好，可以在 papers with code 上的 benckmark 查到 ResNet-50 在 ImageNet-1k 上的 test_acc = 77.15，但是在 JFT-300M 上做预训练之后，再做 transfer 的话，可以达到 test_acc = 87.54。这里又不得不感慨，大力出奇迹，只是这次换成了数据集。。。。。

想了下，还是要放代码的，我这里放一下 train.py:

```
import torch

from utils.eval import calc_acc
from utils.utils import setup_seed, get_dataloader, show_img, predict_batch, set_gpu
from utils.model import define_model, define_optim, start_train
from torch import nn

# 设置哪块显卡可见
device = set_gpu('0, 1')

# 设置随机数种子，使结果可复现
setup_seed(20)

# 数据读取
train_batch = 160
test_batch = 160
EPOCH = 200
trainloader, testloader, classes = get_dataloader(train_batch, test_batch)

# 对训练集的一个batch图片进行展示
show_img(trainloader, classes, batch_size=train_batch)

# 网络结构定义
net = define_model(classes)
net = nn.DataParallel(net)
net.to(device)

# 定义优化器和损失函数
criterion, optimizer = define_optim(net)
net.load_state_dict(torch.load('resnetV1-50-9519-93.pth'))

# 开始模型训练
start_train(net, EPOCH, trainloader, device, optimizer, criterion, testloader)

# 训练后在测试集上进行评测
net.load_state_dict(torch.load('resnet50Cls.pth'))
print(calc_acc(net, testloader, device))

# 进行模型预测
predict_batch(net, testloader, classes, test_batch, device)
```

完整代码的话，我放到 Github了: https://github.com/MaoXianXin/PycharmProjects

补充论文解读: Deep Residual Learning for Image Recognition，主要解决网络加深，模型优化困难问题。
Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously.

解读: 训练深度神经网络是困难的，当然浅层的网络不困难，该篇论文提出残差学习架构来解决该问题。

We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions.

解读: 这里可以理解为引入残差连接。

The depth of representations is of central importance for many visual recognition tasks.

Deep networks naturally integrate low/mid/high-level features and classifiers in an end-to-end multi-layer fashion, and the “levels” of features can be enriched by the number of stacked layers(depth).

解读: 对于很多视觉识别任务来说，网络的深度是非常重要的，可以理解为分别提取到 low/mid/high 三种层次的特征。

Driven by the significance of depth, a question arises: Is learning better networks as easy as stacking more layers ? An obstacle to answering this question was the notorious problem of vanishing/exploding gradients, which hamper convergence from the beginning. This problem, however, has been largely addressed by normalized initialization and intermediate normalization layers, which enable networks with tens of layers to start converging for stochastic gradient descent(SGD) with back-propagation.

解读: 对于验证模型深度越深，网络的性能是否越好之前，这里还存在一个阻碍，就是梯度消失和爆炸，不过这个问题很大程度上可以通过正则初始化以及中间正则化层解决。

When deeper networks are able to start converging, a degradation problem has been exposed: with the network depth increasing, accuracy gets saturated and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training error.

The degradation(of training accuracy) indicates that not all systems are similarly easy to optimize.

解读: 在解决了网络的梯度消失和爆炸之后，我们的网络可以正常的收敛了，但是随着网络加深，出现了准确率饱和以及快速退化的问题，并且这个问题不是由过拟合造成的。我们还可以这样理解，对于一个适当深度的网络，如果你在这个基础之上再添加层数，这个时候你不会获得更好的性能，反而会得到更大的误差。

We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping.

解读: 这个假设是这篇论文的关键，也是提出 residual mapping 的立足点。

To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

疑惑: 还不是特别理解，待后面解决吧。目前的理解就是，优化添加了残差连接的网络比直接优化堆叠起来的非线性层更容易。

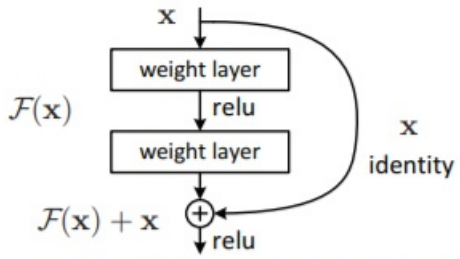

Figure 2. Residual learning: a building block.

In our case, the shortcut connections simply perform identity mapping, and their outputs are added to the outputs of the stacked layers. Identity shortcut connections add neither extra parameter nor computational complexity. The entire network can still be trained end-to-end by SGD with backpropagation, and can be easily implemented using common libraries.

解读: 此处提到的 identity mapping 既不增加额外的参数，也不增加计算复杂度。

We evaluate our method on the ImageNet 2012 classification dataset that consists of 1000 classes. The models are trained on the 1.28 million training images, and evaluated on the 50k validation images. We also obtain a final result on the 100k test images, reported by the test server.

解读: ResNet 模型是在 ImageNet 2012，可以称为 ImageNet-1k 数据集上训练的，有 1.28M 张训练图片，50k 张验证图片，以及 100k 张测试图片。

we also note that the 18-layer plain/residual nets are comparably accurate, but the 18-layer ResNet converges faster.

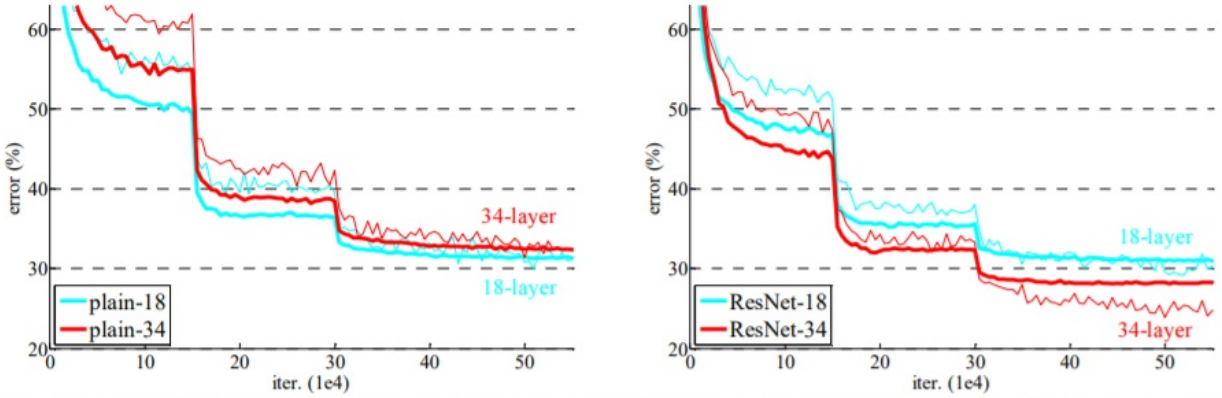

Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

解读: 从上图，确实可以看出 ResNet-18 在初始阶段收敛速度快于 Plain-18。

Bottleneck Architectures: a stack of 3 layers, 1x1, 3x3, and 1x1 convolutions, where the 1x1 layers are responsible for reducing and then increasing(restoring) dimensions leaving the 3x3 layer a bottleneck with smaller input/output dimensions.

The parameter-free identity shortcuts are particularly important for the bottleneck architectures. If the identity shortcut is replaced with projection, one can show that the time complexity and model size are doubled, as the shortcut is connected to the two high-dimensional ends. So identity shortcuts lead to more efficient models for the bottleneck designs.

解读: 此处说的是 Bottleneck 的结构，两端是 1x1，中间是 3x3，同时两端的 dimension 大，中间的 dimension 小，可以节省参数量和计算量

The 50/101/152-layer ResNets are more accurate than the 34-layer ones by considerable margins.

model	top-1 err.	top-5 err.
VGG-16 [41]	28.07	9.33
GoogLeNet [44]	-	9.15
PReLU-net [13]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

Table 3. Error rates (%, **10-crop** testing) on ImageNet validation. VGG-16 is based on our test. ResNet-50/101/152 are of option B that only uses projections for increasing dimensions.

解读: 从上图确实可以看出来 50/101/152 层的 ResNet 准确率比 34 层的高。

We also notice that the deeper ResNet has smaller magnitudes of responses, as evidenced by the comparisons among ResNet-20, 56, and 110. When there are more layers, an individual layer of ResNets tends to modify the signal less.

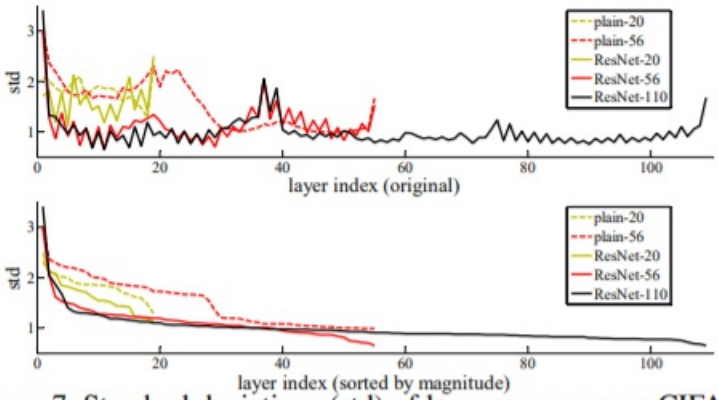

Figure 7. Standard deviations (std) of layer responses on CIFAR-10. The responses are the outputs of each 3×3 layer, after BN and before nonlinearity. **Top**: the layers are shown in their original order. **Bottom**: the responses are ranked in descending order.

解读: 此处展示不同网络的各个层的 magnitudes of responses