Alexnet 详细解读

目录

一、简要介绍

卷积神经网络(CNN)是深度学习领域中十分重要的一个网络框架。第一个经典的 CNN 结构网络是 LeNet5，于 1998 年由 LeCun 提出，它解决了手写数字识别的问题。但在这之后，深度学习领域就沉寂了下来。直到 2012 年，Alexnet 的网络的提出才重新引爆了整个深度学习领域，它为后续的 VGG、GoogLeNet、ResNetCNN 甚至是 R-CNN 等网络定下了基调，更是让 CNN 从此在计算机视觉领域一枝独秀。Alexnet 在 2012 年 ImageNet LSVRC 比赛中夺取了冠军，且准确率远超第二名，它 top-5 的误差率是 15.3%，而第二名是 26.2%。我认真阅读了 Alexnet 的论文原文，尽自己最大的努力将其译为中文，并且运用 latex 进行排版，力求和原文相同。本文则是结合自己的理解对 Alexnet 的解读以及对 ALexnet 网络的实现。

二、数据集

该网络用的是 ImageNet 数据集，关于数据集本身就不多做介绍了，因为论文原文写得十分清楚了。值得注意的就是论文里面提到的误差率(我认为论文对其的解释并不很清楚)，一个叫做 top-1，一个叫做 top-5 。对一个测试图片，若模型预测概率最大的就是正确答案，那么认为预测正确。top-1 错误率就是指模型对一张图片预测的答案不是正确答案的占比（也就是说就是传统意义上的误差率）。对一个测试图片，如果模型预测的概率前五中包含正确答案，即认为模型该图片预测结果正确。那么 Top-5 错误率就是指模型对一张图片预测前五的概率中不包含正确结果的占比。

另外一点就是本文的网络结构没有对数据集进行复杂的预处理，直接像素的原始 RGB 值上训练了网络，它唯一做的事情就是将图片进行了缩放剪裁，使其符合输入网络的维度。

三、四大创新点

一、ReLU 函数

在此之前，网络对激活函数的选择大多数是 sigmoid 或者 tanh，这两个函数十分的经典，但却有致命的缺陷。在网络过大的时候，它们会引起**梯度消失**或是**梯度爆炸**。ReLU 函数不仅不存在这个问题，并且在论文中，作者有提到 ReLU 的训练速度也比 tanh 大概快 6 倍左右。

接下来定性描述一下梯度消失和梯度爆炸，仅以 sigmoid 为例：

梯度消失：根据 sigmoid 的特点，它会将+∞～-∞之间的输入压缩到 0～1 之间。当 input 的值更新时，output 会有很小的更新。又因为上一层的输出将作为后一层的输入，而输出经过 sigmoid 后更新速率会逐步衰减，直到输出层只会有微乎其微的更新。

梯度爆炸：若前面 layer 的梯度通过训练变大，而后面 layer 的梯度指数级增大，这种现象就叫做梯度爆炸(exploding gradient problem)。

二、局部响应归一化

在这篇文章中首次提出了 LRN（局部响应归一化）层，它的原理是仿造生物学上活跃的神经元对相邻神经元的抑制现象（侧抑制）。它会对局部神经元的活动创建竞争机制，使得其中响应比较大的值变得相对更大，并抑制其他反馈较小的神经元，以此来增强模型的泛化能力。原文提到，LRN 将 top-1 与 top-5 误差率分别减少了 1.4% 与 1.2% 。有趣的是，在后来的 VGG 中，作者认为 LRN 并没有什么用，它不能在 ILSVRC 数据集上提升性能，只是会导致更多的消耗更多的内存和计算时间而已，这种方法后来也就很少使用了。

三、Dropout

训练时使用 **Dropout** 随机忽略一部分神经元，以避免模型过拟合。Dropout 虽有单独的论文论述，但是 AlexNet 将其实用化，通过实践证实了它的效果。在 Alexnet 中主要是最后几个全连接层使用了 Dropout。

四、重叠池化

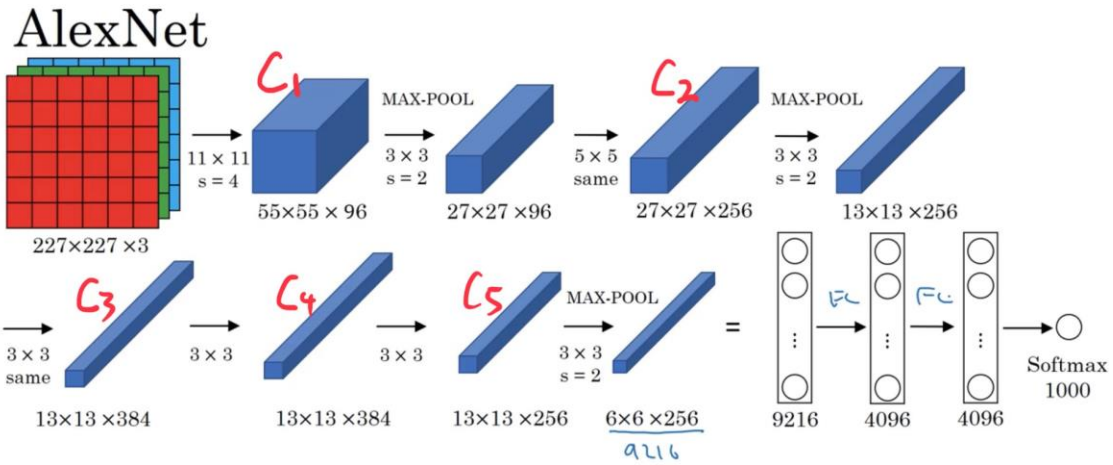
在 CNN 中使用重叠的**最大池化**。此前 CNN 中普遍使用平均池化，AlexNet 全部使用最大池化，避免平均池化的模糊化效果。并且 AlexNet 中提出让步长比池化核的尺寸小，这样池化层的输出之间会有重叠和覆盖，提升了特征的丰富性。论文中提到这个方案分别降低了 top-1 0.4% 和 top-5 0.3% 的误差率，而且会使得网络更加不容易过拟合。

四、网络结构的分析

在原论文中，因为作者使用了两个 GPU 进行训练，所以论文中总体结构一部分的原图并不能轻易理解。好在我们如今的硬件发展已经到使用单 GPU 就可以实现，所以也没有必要去了解那么复杂的结构，我们可以将其放在一个 GPU 上看网络结构。

值得注意的是，其实输入图像的尺寸不应该是 224x224x3，而是 227x227x3（我尝试使用 224x224x3 进行计算，发现边界填充的结果是小数，就算向下取整，第二层也是 54x54x96，和原文第二层不一致，学界也普遍认为这是一个原文的错误）。

这里借用一下吴恩达上课时的图片来说明网络结构



- AlexNet 首先用一张 227×227×3 的图片作为输入。
- C1 第一层我们使用 96 个 11×11 的过滤器，步幅为 4，由于步幅是 4，因此尺寸缩小到 55×55 。然后用一个 3×3 的过滤器构建最大池化层，f=3,s=2,尺寸缩小为 27×27×96。
- C2 接着再执行一个 5×5 的 same 卷积， p=3,s=1，输出是 27×27×256。然后再次进行最大池化，尺寸缩小到 13×13。
- C3 再执行一次 same 卷积，p=1,s=1，得到的结果是 13×13×384， 384 个过滤器。
- C4 再做一次 same 卷积，p=1,s=1，得到的结果是 13×13×384， 384 个过滤器。
- C5 再做一次 same 卷积，p=1,s=1，得到的结果是 13×13×256。最后再进行一次最大池化，尺寸缩小到 6×6×256。
- 6×6×256 等于 9216，将其展开为 9216 个单元，然后接入全连接层。最后使用 softmax 函数将结果分为 1000 类。

最后还是根据原文的描述，说明一下每个卷积核的参数情况 C1：96x11x11x3 （卷积核个数/高/宽/深度） C2：256x5x5x48（卷积核个数/高/宽/深度） C3：384x3x3x256（卷积核个数/高/宽/深度） C4：384x3x3x192（卷积核个数/高/宽/深度） C5：256x3x3x192（卷积核个数/高/宽/深度）

在这里 C2，C4，C5 卷积核的深度有些奇怪，它们不等于前一个卷积核的个数。这主要是原文使用两个 GPU 造成的，这几层的核分布在两个 GPU 上，后一个卷积层只连接同一个 GPU 上的上一个网络，所以造成卷积核深度减半了。

五、网络优化与未来展望

一、个人理解

在我阅读这篇论文的时候，其实还有一个技术也很具创新点，那就是使用了 CUDA 利用两个 GPU 并行计算，我觉得这是作者的无奈之举，因为 GPU 性能确实差。技术的更迭对于今天的我来说已经不需要在考虑这个问题，因为我们现在的 GPU 厂商推出了更多更强的 GPU，并且它们也对 GPU 在深度学习方面做了足够的优化。我就不再需要从硬件上面去考虑网络的运行了。关心网络中的优化算法和结构才是我应该做的。

二、网络优化

该论文为了简化实验没有使用任何无监督的预处理手段，作者也在论文中提过这一点。在我看来，在进行训练之前，可以先使用 PCA 和白化对图片进行处理（其实论文中也使用了 PCA，不过是在进行数据增强），达到降维的效果，这样图像的特征会更加明显，学习起来也就更加容易。

论文在开始时提到，难以获得带标签的大型数据集，这是阻止其性能上升的一个因素。然而时至今日，随着深度学习的飞速发展，自学习的方面取得了很大的进步，并不一定非要带标签的数据集才可以进行训练，完全可以把自编码器放在网络的前面，利用逐层贪婪预训练的方法来进行特征提取，再输入网络进行训练，这样便会有更大的数据集，从而实现对网络性能的提升。

增加网络的深度绝对是很好的提高性能的方法。完全可以利用残差网络来构建更深的网络结果，从而可以达到很好的效果。Resnet 就是这样做的，Resnet-152 的 top-1 误差率已经低至 22%左右，相比 alexnet 又降低了一倍了。

三、未来展望

Alexnet 虽只是一篇 CNN 的开山之作，但对未来已经造成了长远的影响。我们熟知的阿尔法狗、现在流行的人脸识别，其核心也使用了 CNN。在当今，CNN 的后代例如 RCNN、Fast-RCNN、Mask-RCNN 已经相继出世，它们不仅能够进行图像分类，甚至已经能够进行语义分割，这些其实就是未来自动驾驶的基础。相信随着以后技术的发展，我们还能迎来更多更强的网络。

六、网络的实现

一、数据集

在读完这篇论文之后，我在 github 上面找到了使用 imagenet 的具体实现。可是，对我来说 imagenet 这个数据集太大了，根本不是我手上的机器能够承受的，所以最后还是决定使用 cifar-10 数据集做一个简单的 Alexnet 网络。

CIFAR-10 一共包含 10 个类别的 RGB 彩色图片：飞机（ airplane ）、汽车（ automobile ）、鸟类（ bird ）、猫（ cat ）、鹿（ deer ）、狗（ dog ）、蛙类（ frog ）、马（ horse ）、船（ ship ）和卡车（ truck ）。图片的尺寸为 32 × 32 ，数据集中一共有 50000 张训练图片和 10000 张测试图片。

在具体实现时使用了 opencv 的 resize 方法将图片尺寸放大至 227 × 227。

二、具体实现

因为本人水平问题，还不足以独立实现 Alexnet，所以实现是参考了网上一大神的代码。该实现是基于 tensorflow 这个著名框架，我细细的读完了整个代码，收获颇丰，同时也对 Alexnet 有了更深的理解，也更加熟悉了 tensorflow。

我在极客云这个平台上面使用 1080 Ti 跑的网络，一共训练了 35 轮左右，花了大概三个小时，最后训练集上准确率达到了 94%，测试集上是 67%左右。

这个结果说明网络过拟合了,我想有以下几点原因:

- 我想应该是我使用的数据集的原因, 把 32 × 32 放大至 227 × 227 不是一个合理的选择
- CIFAR-10 数据集比较小, 数据量的不足是导致过拟合的原因之一
- 这个网络本来是针对 ImageNet 那种大型数据集的,我用了 CIFAR-10,可能训练集的数量级要小于模型的复杂度也是原因之一

解决方案:

- 调小模型复杂度,使其适合自己训练集的数量级（缩小宽度和减小深度）
- 更换数据集,使用更大的数据集进行训练

解决的过拟合的难度并不大,但是我最后还是没能做到,主要原因不是没有理论支撑,是因为在手上的设备不行的情况下,使用极客云训练网络是一件比较费钱的事情,所以也就没能在进行下去了。

七、总结

真的要很感谢老师给的这一次翻译论文的机会，不然我也不会花那么多精力去认真阅读这一篇经典之作，这真的让我受益匪浅。我这次的收获如下：

- 通过这一篇论文真的大大增加了我对 CNN 的理解，并且让我学习到了那些经典的优化技术。
- 通过参考别人的代码进行实现的过程也让我进一步学习了 tensorflow 这个神经网络框架。

- 对论文的排版需求又让我学习了 Latex 的语法，相信对以后写论文帮助极大。
- 英文翻译为中文的过程中也让我英语有所精进。

虽然这篇 alexnet 结束了，不过我相信这应该是我阅读论文的开始，因为我后面还打算将 VGG、ResNet 等等经典论文都读了，以期有进一步的提升。