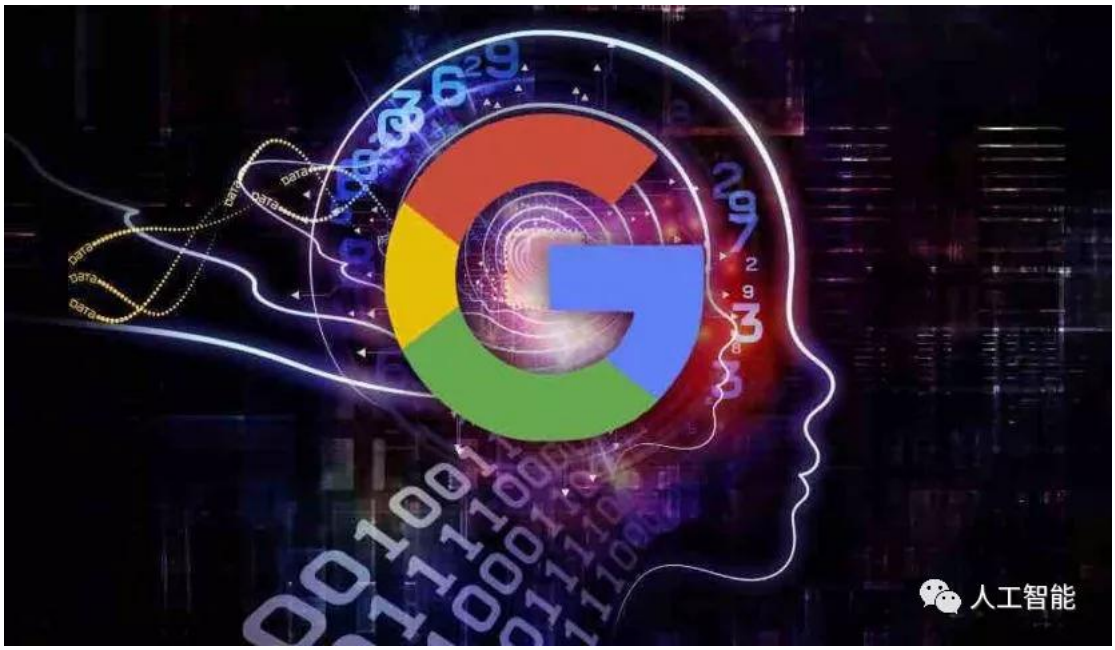
谷歌 MorphNet：让你的神经网络更小但更快

人工智能2019-04-20



深度神经网络（DNN）在解决图像分类、文本识别和语音转换等实际难题方面具有显著的效果。然而，为一个给定的问题设计一个合适的 DNN 体系结构仍然是一个具有挑战性的任务。考虑到架构可能需要巨大的搜索空间，从头开始为特定的应用程序设计一个网络在计算资源和时间方面花销可能非常大。神经网络架构搜索和 AdaNet 等方法利用机器学习来搜索设计空间，以便找到改进架构的方法。另一种选择是将现有的体系结构用于类似的问题，并一次性为手头的任务进行优化。

正对这个问题，谷歌 AI 发布了一篇博文讨论了 MorphNet。MorphNet 是一种复杂的神经网络模型细化技术，它采用了上面说的第二种方法。本文对 MorphNet 的解释是：「深度神经网络的快速、简单的资源受限结构学习」。MorphNet 以现有的神经网络为输入，生成一个更小、更快、性能更好的新神经网络，以适应新的问题。

我们已经将这项技术应用于「Google-scale」问题，以设计更小、更准确的生产服务网络。而且，现在我们已经向社区开放了 MorphNet 的 TensorFlow 实现，这样你就可以使用它来提高你的模型的效率。

它是如何工作的

MorphNet 通过收缩和扩展阶段的循环优化神经网络。在收缩阶段，MorphNet 识别效率低下的神经元，并利用稀疏正则化器将其从网络中删去，这样网络的总损失函数就包含每个神经元的成本。然而，MorphNet 并没有对每个神经元使用一样的成本，而是根据目标资源计算神经元成本。随着训练的进行，优化器在计算梯度时会意识到资源成本，从而了解哪些神经元是是节省资源的，哪些神经元可以被移除。

例如，考虑 MorphNet 如何计算神经网络的计算成本，以触发器为例。为了简单起见，让我们考虑一个用矩阵乘法表示的神经网络层。在这种情况下，层有 2 个输入（xn），6 个权重（a，b，…，f）和 3 个输出（yn；神经元）。使用标准教科书中的行和列相乘的方法，可以计算出评估该层需要 6 次相乘。

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} a & b \\ 0 & d \\ e & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \qquad \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} a & b \\ \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

神经元的计算成本

MorphNet 将其计算为输入计数和输出计数的乘积。注意，尽管左边的例子显示了两个为 0 的权重值，我们仍然需要执行所有的乘法来评估这个层。然而，中间的例子显示了结构化稀疏性，其中神经元 yn 中的所有行权重都为 0。MorphNet 识别出这个层的乘法数从 6 减少到 4，于是这个层的新输出计数是 2。利用这一思想，MorphNet 可以确定网络中每个神经元的增量成本 ,从而生成一个更有效的模型(右边)，其中神经元 y3 已经被移除。

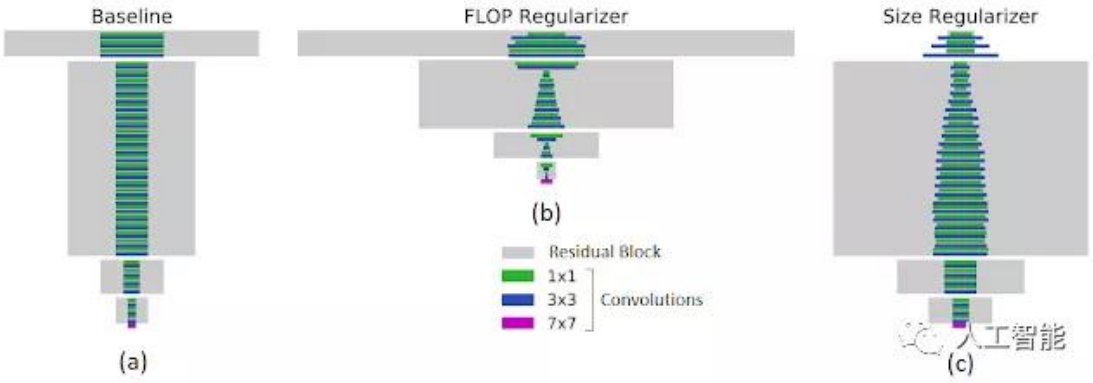
在展开阶段，我们使用宽度乘数来均匀地扩展所有的层大小。例如，如果我们扩大 50%，那么一个以 100 个神经元开始并缩小到 10 个的低效率层只会扩大到 15 个，而一个只缩小到 80 个神经元的重要层可能会扩大到 120 个，并有更多的资源来工作。也就是将计算资源从网络中效率较低的部分重新分配到可能更高效的部分。

在缩减阶段之后，人们可以停止 MorphNet，只需缩减网络以满足更严格的资源预算。这会导致在给定目标成本的时候网络效率更高，但有时也会导致精度下降。另外，用户还可以完成扩展阶段，该阶段将与原始目标资源成本相匹配，但提高了准确性。稍后我们通过一个示例来介绍这个的完整实现过程。

为什么是 MorphNet？

MorphNet 提供了四个关键的有价值的主张：

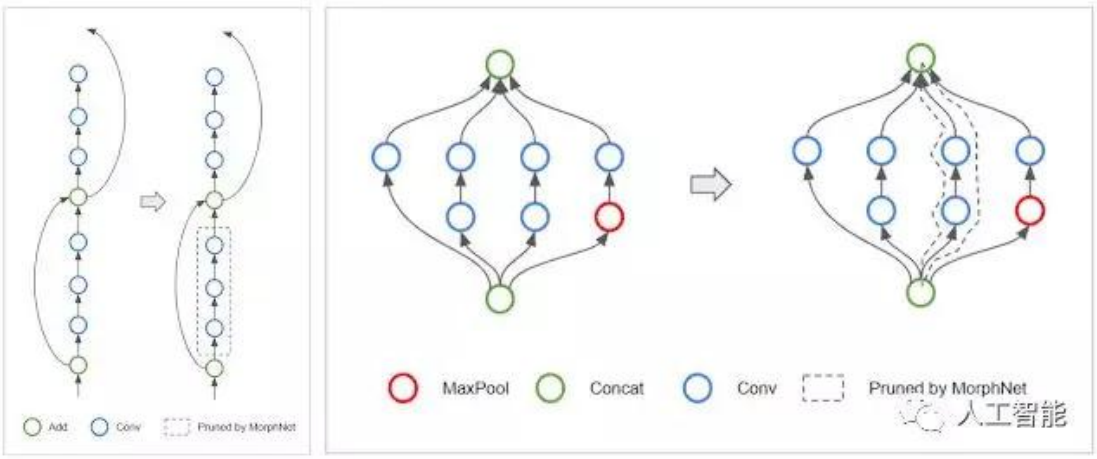
有针对性的正则化：与其他稀疏的正则化方法相比，MorphNet 采取的正则化方法目的性更强。尤其是，MorphNet 方法进行更好的稀疏化的目的是减少特定的资源。这可以更好地控制由 MorphNet 生成的网络结构，根据应用领域和相关约束，MorphNet 可以有明显的不同。例如，下图的左侧面板显示了一个基线网络，该网络具有在 JFT 上训练的常用 ResNet-101 体系结构。当以触发器（中间图，触发器减少 40%）或模型大小（右图，权重减少 43%）为目标时，MorphNet 生成的结构有很大不同。在优化计算成本时，较低层网络中的高分辨率神经元比低分辨率神经元更容易受到修剪。当模型尺寸较小时，在修剪权衡上正好相反。



MorphNet 有针对性的正则化。矩形宽度与层中的通道数成正比。底部的紫色条是输入层。左图：基线网络用作 MorphNet 的输入。中图：输出应用触发器调节器。右图：输出应用大小调整器。

MorphNet 是为数不多的能够针对特定参数进行优化的解决方案之一。这使它能够针对特定实现的参数。例如，可以通过结合特定于设备的计算时间和内存时间，将延迟作为一阶优化参数。

拓扑变形：当 MorphNet 学习每层神经元的数量时，算法在一个层中稀疏所有神经元的过程中可能会遇到一种特殊的情况。当一个层有 0 个神经元时，通过切断网络中受影响的分支，可以有效地改变网络的拓扑结构。例如，当遇到 ResNet 体系结构时，MorphNet 可能保留 skip-connection，但删除残差块，如下左图所示。对于 Inception 样式的架构，MorphNet 可能会删除整个平行的塔，如右图所示。



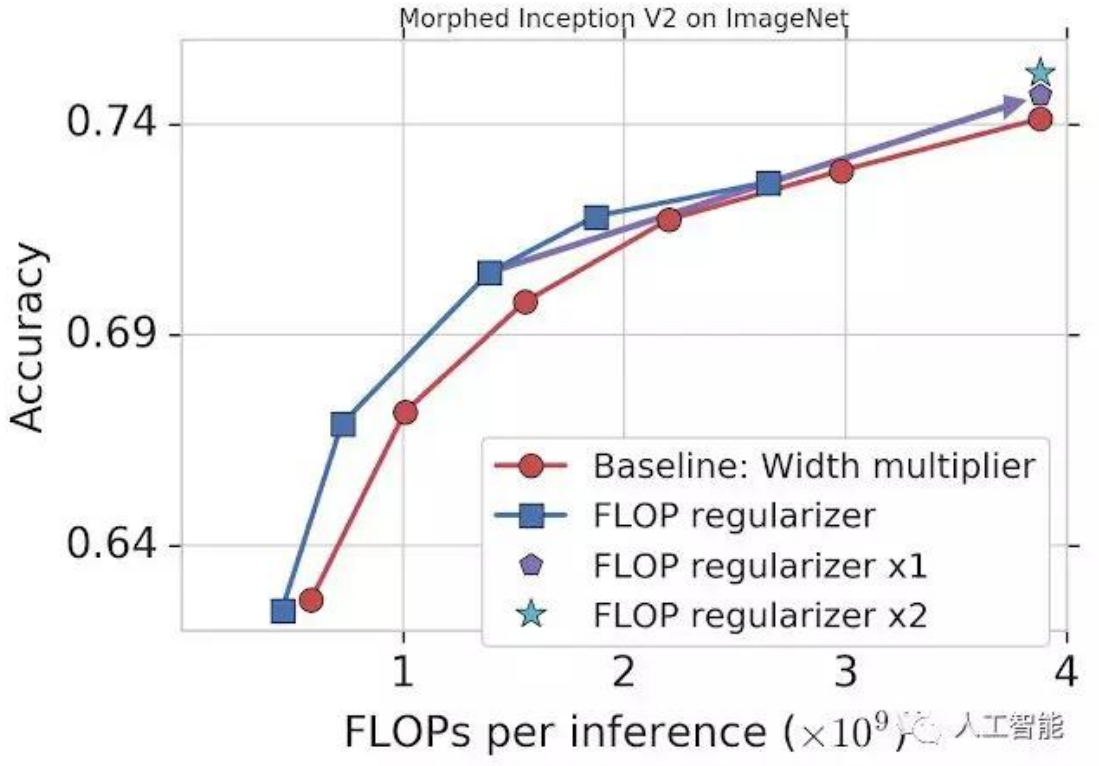
左图：MorphNet 可以删除 ResNet 样式网络中的残差连接。右图：它还可以删除 Inception 样式的网络中的平行塔。

可扩展性：MorphNet 在一次训练中学习新的结构，在培训预算有限时，它是一个很好的方法。MorphNet 也可以直接应用于昂贵的网络和数据集。例如，在上面的比较中，MorphNet 直接应用于 ResNet-101，而它最初是在 JFT 上花费了 100 个 GPU 月训练的。

可移植性：MorphNet 产生的网络是「可移植的」，从这个意义上说，它们是打算从头开始重新训练的，并且权重与体系结构学习过程无关。你不必担心复制检查点或遵循特殊的训练规则，而只需像平时一样训练你的新网络！

变形网络

作为一个演示，我们将 MorphNet 应用于在 ImageNet 上通过目标定位 FLOPs 训练的 Inception V2（见下文）。基线方法是使用一个宽度倍增器，通过均匀地缩小每个卷积（红色）的输出数量来权衡精度和触发器。MorphNet 方法的目标是直接 FLOPs，并在缩小模型时产生更好的权衡曲线（蓝色）。在这种情况下，与基线相比，触发器成本降低了 11% 到 15%，而精确度相同。



MorphNet 应用于 ImageNet 上的 Inception V2。单独使用 FLOP 正则化器（蓝色）可将性能相对于基线（红色）提高 11-15%。在一个完整的周期中，正则化器和宽度乘法器在相同的成本（「x1」；紫色）下提高了精度，并在第二个周期（「x2」；青色）持续改进。

此时，您可以选择一个 MorphNet 网络来满足较小的 FLOP 预算。或者，您可以通过将网络扩展回原始的 FLOP 成本来完成这个周期，从而在相同的成本（紫色）下获得更好的准确性。再次重复变形网缩小\扩展循环会导致另一个精度增加（青色），使总精度增加 1.1%。

结论

我们已经将 MorphNet 应用到了谷歌的几个量产级图像处理模型中。使用 MorphNet 可以在质量几乎没有损失的情况下显著减少模型大小。我们邀请您尝试 MorphNet。可以在这里找到开源 TensorFlow 实现方法，还可以阅读 MorphNet 论文了解更多详细信息。

via：https://ai.googleblog.com/

- END -