这个Note是玥儿在研究深度学习在手机端应用过程中做的笔记

其中会涉及到很多其他论坛的知识内容

只是随手笔记，所以不是非常工整，大家不要介意

A, 安装CUDA

https://www.leiphone.com/news/201711/GCh0IBszXrxP1iHU.html

B，如何让应用在手机端呢？

BinaryConnect：Training Deep Learing Neural Networks with binary weight during propagations），这篇论文引起了广泛关注，开启了崭新的二值化网络时代。论文中提出了 BinaryConnect 算法的关键在于仅在前向传播和反向传播中对权重进行二值化 1 或-1，而在参数更新过程保持权重的全精度（即仍为浮点数），这样的做法可以省去接近三分之二的矩阵运算，训练时间和内存空间都得到了大幅度优化，同时，BinaryConnect 在 MNISIST,CIFAR-10 和 SVHN 图像分类数据集上的实验效果可以达到当时世界领先水平。

在此基础上，Matthieu 和 Itay 随后联合发表的论文（Binarized Neural Networks：Training Networks with Weights and Activations Constrained to +1 or -1）提出了更完善的网络模型——BinaryNet，将权值和隐藏层激活值同时进行二值化，并利用 xnorcount 和 popcount 运算操作代替网络中传统的算术运算。这个算法在常用图像数据集上的模型二值化实验也比较成功，可以减少约 60% 的计算复杂度，甚至可以在保证分类准确率的情况下，减少七倍的 GPU 运行时间。同时，实验团队也公开了在 CUDA、Theano 以及 Torch 上的代码，不过很可惜的是，这个算法并没有在如 ImageNet 的大数据集上证明精度是否可以维持。在尽可能减小模型准确率损失的情况下，BinaryNet 的出现通过缩减模型大小，简化运算难度对算法进行加速。这使得深度网络部署于移动端的前景初见曙光。

值得一提的是，随着神经网络技术的蓬勃发展，许多教授、学者投身工业界，像 Matthie Courbariaux 现已投身 Google，并负责在 TensorFlow 框架中实现对深度模型的量化任务。不同于 Matthie 在论文中的二值化概念 (即不丢失模型准确率，只压缩模型大小), 实际投入应用的量化更适合被理解为离散化。一般来说，在训练神经网络的时候，要对权重做一些微小的调整，而这些微小的调整需要浮点精度才能正常工作，而低精度计算会被网络当做一种噪声。深度网络的一个奇妙之处就在于它可以很好地应对输入噪音，因为网络可以把低精度计算当做一种噪声，这使得量化后的网络在具备较少信息的数值格式下，仍能产生精确的结果。

2016 年 3 月，Mohammad Rastegari 等人在论文 (XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks) 中首次提出了 XNOR-Net 的概念。这篇论文旨在利用二值化操作寻找到最优的简化网络，并分别介绍了两种有效的网络：Binary-Weight-Networks 和 XNOR-Networks。Binary-Weight-Networks 是对 CNN 中所有的权重做近似二值化，可以节省 32 倍的存储空间。而且，由于权重被二值化，卷积过程只剩加减算法，不再包括乘法运算，可以提高约两倍的运算速度，这促使 CNN 可以在不牺牲准确率的情况下在小存储设备上使用，包括便携式设备。Binary-Weight-Networks 区别于 BinaryNet 的地方在于它进行二值化的方法和网络结构。

XNOR-Networks 算法则是对 CNN 中所有的权重和输入同时做近似二值化，如果卷积运算中的所有操作数都是二进制的，那么两个二进制向量的点乘就可以等同于同或运算和位运算。作者在这篇文章里主要有两个贡献：一是引入比例因子，大幅度提升精度；二是对典型常规的 CNN 构成进行改动。

 XNOR-Net 算法的基本思路如下：https://www.jiqizhixin.com/articles/2017-02-20-3

Step1：定义一个 L 层的 CNN 结构，使用三个元素 I,W,\* 来表示，I 表示卷积输入，W 表示滤波器，\*表示卷积算子；利用比例因子α帮助二值化滤波器去近似全精度滤波器权重，利用比例因子β帮助二值化的输入去近似全精度输入值。这有点类似于批规范化（Batch Normalization）中的仿射参数（Affine Parameters），但不同的是，这里不是通过为网络学习获得的，而是通过计算平均值得到的。在探索计算比例因子β的时候，要对每一次卷积产生的子张量都计算一个β，这一步骤会产生很多冗余的计算。为了降低计算量，作者把输入的所有 channels 计算一个绝对值的平均值矩阵 A，然后通过一个二维的滤波器 k 和 A 的卷积生成 K，那么 K 就包含了针对输入 I 在所有子张量上的比例因子。通过这样一系列数学推导，输入 I 与权重 W 的二值化卷积可以被近似。

C，读书笔记

论文：[A Survey of Model Compression and Acceleration for Deep Neural Networks](https://arxiv.org/pdf/1710.09282.pdf)

 深度神经网络模型压缩与加速综述

**摘要**--深度卷积神经网络（CNN）目前在很多视觉识别任务中有非常良好的表现。然而，目前的深度卷积神经网络模型非常耗费计算资源和内存，面临着在终端部署和低延迟需求场景下难以应用的问题。因此，一种很自然的解决方案就是在保证分类准确率不显著下降的前提下对深度卷积神经网络进行压缩和加速。近年来，该领域有很大发展。我们将在本论文中介绍近期压缩和加速 CNN 模型的先进技术。这些技术可以大致分为四类：参数裁剪和共享（parameter pruning and sharing）、低秩分解（low-rank factorization）、迁移/压缩卷积滤波器（transfered/compact convolutional filter）和知识精炼（knowledge distillation）。参数修剪和共享的方法将在论文开头详细描述**，**其他几类方法也都会在文中介绍。我们对每一类方法的性能、相关应用、优势和缺陷等进行独到的分析。然后本文将介绍几个最近的其他成功方法，如动态网络和随机深度网络（stochastic depths network）。之后，我们将研究评估矩阵（evaluation matrix）——用于评估模型性能和近期基准的主要数据集。最后，我们总结并讨论了现有的挑战和可能的发展方向。

**1 参数裁剪和共享**

可以进一步分为三类：

**(1)模型量化和二进制化**

通过减少权重所占存储空间来压缩网络。

      网络量化通过减少表示每个权重所需的比特数来压缩原始网络。Gong et al. [6] 和 Wu et al. [7] 对参数值使用 K 均值标量量化。Vanhoucke et al. [8] 展示了 8 比特参数量化可以在准确率损失极小的同时实现大幅加速。[9] 中的研究在基于随机修约（stochastic rounding）的 CNN 训练中使用 16 比特定点表示法（fixed-point representation），显著降低内存和浮点运算，同时分类准确率几乎没有受到损失。

       S. Han et al. [10] 提出的方法是首先修剪不重要的连接，重新训练稀疏连接的网络。然后使用权重共享量化连接的权重，再对量化后的权重和码本（codebook）使用霍夫曼编码，以进一步降低压缩率。如图 1 所示，该方法首先通过正常的网络训练来学习连接，然后再修剪权重较小的连接，最后重新训练网络来学习剩余稀疏连接的最终权重。

**缺陷**：此类二元网络的准确率在处理大型 CNN 网络如 GoogleNet 时会大大降低。另一个缺陷是现有的二进制化方法都基于简单的矩阵近似，忽视了二进制化对准确率的影响。针对这个问题，[16]提出一个=具有对角Hessian近似的牛顿算法(a proximal Newton algorithm with diagonal Hessian approximation )，可以直接最小化关于二进制权重的损失，[17]通过随机二值化权重并将隐藏状态计算中的乘法转换为符号变化来减少训练阶段中浮点乘法的时间。

**(2)剪枝和共享**

     网络剪枝和共享已经被用于降低网络复杂度和解决过拟合问题。有一种早期应用的剪枝方法称为偏差权重衰减法( the Biased Weight Decay)[18]，其中最优脑损伤(The Optimal Brain Damage)[19]和最优脑手术(the Optimal Brain Surgeon [20])的方法基于损失函数的 Hessian 矩阵减少连接的数量，他们的研究表明这种剪枝方法的精确度比基于重要性的剪枝方法更高。

      这个方向最近的趋势是在预先训练的CNN模型中修剪冗余的，非信息量的权重。 例如，Srinivas和Babu [21]研究了神经元之间的冗余，并提出了一种无数据修剪方法去除多余的神经元。Han er al. [22]建议减少整个网络中的参数和操作总数。 Chen et al. [23]提出了一个HashedNets模型，该模型使用低成本的散列函数将权重分组为哈希表进行参数共享。文献[10]中的深度压缩方法去掉冗余连接并量化权重，然后使用霍夫曼编码对量化权重进行编码。 在[24]中，提出了一种基于软重量共享的简单正则化方法，其中包括在一个简单（重新）训练过程中的量化和修剪。 值得注意的是，上述修剪方案通常会在CNN中产生连接修剪。

    越来越多的学者对训练稀疏性限制的紧凑型CNN(compact CNNs with sparsity constraints )感兴趣，这些稀疏性约束通常在优化问题中作为L0或L1范数被引入。[25]中的工作对卷积滤波器施加了组稀疏约束(group sparsity constraint )，以实现结构化的脑损伤(structured brain Damage )，即以分组方式修剪卷积核。[26 ]在训练阶段引入了一组神经元上的稀疏正则化器，学习紧凑CNNs，可以减少卷积核的数量。[27]在每层上增加了一个结构稀疏正规化器，以减少普通的滤波器、通道甚至层。在滤波器级修剪中，所有上述工作都使用了L2 L1范数正则化器。[28]中的工作使用l1范数来选择和修剪不重要的过滤器。

**缺陷**：剪枝和共享方法存在一些潜在的问题。首先，若使用了 L1 或 L2 正则化，则剪枝方法需要更多的迭代次数才能收敛，此外，所有的剪枝方法都需要手动设置层的敏感度，即需要精调超参数，在某些应用中会显得很冗长繁重。

**(3)设计结构化矩阵**

       在只包含全连接层的网络中，参数数量可达到10亿[4]，因此，研究全连接层的参数冗余是非常重要的，这通常是内存消耗方面的瓶颈所在。如果一个 m x n 阶矩阵只需要少于 m×n 个参数来描述，就是一个结构化矩阵（structured matrix）。通常这样的结构不仅能减少内存消耗，还能通过快速的矩阵-向量乘法和梯度计算显著加快推理和训练的速度。遵循这个方向，[29] [30]中的工作提出了一种基于循环预测的简单而有效的方法，同时保持了很低的错误率。给定一个向量r=针对核张量(kernel tensors ),[38]提出了典型的张量分解(Canonical Polyadic (CP) decomposition )，他们的工作使用非线性最小二乘法来计算CP分解。在[39]中，提出了一种新的从一开始就用于训练低秩约束CNNs的低秩张量分解算法，它使用BN来转换内部隐藏单元的激活。

        通常，[39]提出的CN和BN分解方案可以在一开始就训练CNNs，然而他们之间没有什么区别，例如，在CP分解中找到最好的低秩近似并不合适，有时可能不存在最佳秩K（K是秩数）近似。 而对于BN方案，分解总是存在的。我们还比较了表II中两种方法的性能。

https://img-blog.csdn.net/20180509161915811，循环矩阵https://img-blog.csdn.net/20180509162016763可以定义为：

 通过实际的加速和压缩率来衡量它们的性能。 可以看到，BN方法可以实现略好的加速率，而CP版本提供更高的压缩率。

       全连接层可视为2D矩阵，因此上述方法也可应用于此。关于在完全连接的层中利用低秩的几个经典work，例如Misha et al. [40]使用低秩方法减少了深层模型中的动态参数数量。 [41]探索DNN中最终加权层的低阶矩阵分解以进行声学建模

**缺陷**：低秩方法很适合模型压缩和加速，该方法补充了深度学习的近期发展，如 dropout、修正单元（rectified unit）和 maxout。但是，低秩方法的实现并不容易，因为它涉及计算成本高昂的分解操作。另一个问题是目前的方法逐层执行低秩近似，无法执行非常重要的全局参数压缩，因为不同的层具备不同的信息。最后，分解需要大量的重新训练来达到收敛。

**3 迁移/压缩卷积滤波器**

设计特殊结构的卷积滤波器以减少存储和计算的复杂度。

     CNN对输入图像的特征提取具有一定的旋转平移不变的性质，这是能够训练非常深的网络的关键，不会出现严重的过拟合，虽然目前还缺少一个强有力的理论，但有大量实证支持，即平移不变性质和参数共享对良好的预测结果是很重要的。使用转移卷积滤波(transferred convolutional filters )器来压缩CNN模型的想法是由[42]的工作所推动的，它引入了等变群理论(equivariant group theory )。使 x 作为输入，Φ(·) 作为网络或层，T (·) 作为变换矩阵。则等变概念可以定义.

<https://blog.csdn.net/u014380165/article/details/77731595>

使用变换矩阵 T (·) 转换输入 x，然后将其传送至网络或层Φ(·)，其结果和先将 x 映射到网络再变换映射后的表征结果一致。根据该理论，将变换矩阵应用到层或滤波器Φ(·) 来对整个网络模型进行压缩是合理的。

**缺陷**：将迁移信息应用到卷积滤波器的方法需要解决几个问题。首先，这些方法的性能可与宽/平坦的架构（如 VGGNet）相媲美，但是无法与较窄/特殊的架构（如 GoogleNet、Residual Net）相比。其次，迁移假设有时过于强大以致于无法指导算法，使得在某些数据集上的结果不稳定。

**4 知识精炼**

        训练一个更加紧凑的神经网络以再现大型网络的输出结果

       据我们所知，Caruana et al.[49] 首先提出利用知识迁移（KT）来压缩模型。他们通过集成强分类器标注的伪数据训练了一个压缩模型，并再现了原大型网络的输出结果。然而他们的工作仅限于浅层网络。这个想法近来在 [50] 中扩展为知识精炼（Knowledge Distillation/KD），它可以将深度和宽度的网络压缩为浅层模型，该压缩模型模仿了复杂模型所能实现的功能。KD 的基本思想是通过软 softmax 学习教师输出的类别分布而降大型教师模型（teacher model）的知识精炼为较小的模型。  
[51] 中的工作引入了 KD 压缩框架，即通过遵循学生-教师的范式减少深度网络的训练量，这种学生-教师的范式即通过软化教师的输出而惩罚学生。该框架将深层网络（教师）的集成压缩为相同深度的学生网络。为了完成这一点，学生学要训练以预测教师的输出，即真实的分类标签。尽管 KD 方法十分简单，但它同样在各种图像分类任务中表现出期望的结果。[52]中的工作旨在通过利用深度神经网络来解决网络压缩问题。 它提出了一种方法来训练薄而深的网络，称为FitNets，以压缩宽而浅（但仍然很深）的网络。 该方法扩展了思路，以允许更薄更深的学生模型。 为了从教师网络的中间表现中学习，FitNet使学生模仿了教师的全部特征地图。 但是，这样的假设太严格了，因为师生的能力可能差别很大。 在某些情况下，FitNet可能会对性能和收敛性产生不利影响。 所有上述方法都在MNIST，CIFAR-10，CIFAR-100，SVHN和AFLW基准数据集上进行了验证，仿真结果表明这些方法匹配或超过教师的表现，而要求显着减少参数和乘法。

**缺点**：基于 KD 的方法能令更深的模型变得更加浅而显著地降低计算成本。但是也有一些缺点，例如 KD 方法只能用于具有 Softmax 损失函数分类任务，这阻碍了其应用。另一个缺点是模型的假设有时太严格了，以至于其性能有时比不上其它方法。

**讨论与挑战**

深度模型的压缩和加速技术还处在早期阶段，目前还存在以下挑战：

1、大多数目前的顶尖方法都建立在设计完善的 CNN 模型的基础上，这限制了改变配置的自由度(例如，网络结构和超参数)。为了处理更加复杂的任务，还需要更加可靠的模型压缩方法。

2、剪枝是一种压缩和加速 CNN 的有效方式。目前大多数的剪枝技术都是以减少神经元之间的连接设计的。另一方面，对通道进行剪枝可以直接减小特征映射的宽度并压缩模型。这很有效，但也存在挑战，因为减少通道会显著地改变下一层的输入。确定这类问题的解决方式同样很重要。

3、正如之前所提到的，结构化矩阵和迁移卷积滤波器方法必须使模型具有人类先验知识，这对模型的性能和稳定性有显著的影响。研究如何控制强加先验知识的影响是很重要的。

4、知识精炼(knowledge distillation/KD)方法有很多益处比如不需要特定的硬件或实现就能直接加速模型。开发基于 KD 的方法并探索如何提升性能仍然值得一试。

5、多种小型平台(例如，移动设备、机器人、自动驾驶汽车)的硬件限制仍然是阻碍深层 CNN 扩展的主要问题。如何全面利用有限的可用计算资源以及如何为这些平台设计特定的压缩方法仍然是个挑战。

参考机器之心<https://www.jiqizhixin.com/articles/2017-10-29>

D，SoftMax

一：神经网络中的损失函数

cnn进行前向传播阶段，依次调用每个Layer的Forward函数，得到逐层的输出，最后一层与目标函数比较得到损失函数，计算误差更新值，通过反向传播逐层到达第一层，所有权值在反向传播结束时一起更新。

loss layer 是CNN的终点，接受两个Blob作为输入，其中一个是CNN的预测值，另一个是真实标签。损失层则将这两个输入进行一系列运算，得到当前网络的损失函数（Loss Function），一般记做L(θ）其中θ是当前网络权值构成的向量空间。机器学习的目的是在权值空间中找到让损失函数L(θ） 最小的权值θ（opt），可以采用一系列最优化方法（如SGD方法）逼近权值θ（opt）。

损失函数是在前向传播计算中得到的，同时也是反向传播的起点。

二：Softmax函数，假设有K个类别，Softmax计算过程为：

其中，j=0,1,2,3,4,5,...,K-1下面图更直观：softMax的结果相当于输入图像被分到每个标签的概率分布，该函数是单调增函数，即输入值越大，输出也就越大，输入图像属于该标签的概率就越大。

神评论：SVM只选自己喜欢的男神，Softmax把所有备胎全部拉出来评分，最后还归一化一下

对softmax的结果计算交叉熵分类损失函数为：

取log里面的值就是这组数据正确分类的Softmax值，它占的比重越大，这个样本的Loss也就越小，这种定义符合我们的要求.

三：wiki百科对softmax函数的定义：

In mathematics, the softmax function, or normalized exponential function,[1]:198 is a generalization of the logistic function that "squashes" a K-dimensional vector {\displaystyle \mathbf {z} }\mathbf {z} of arbitrary real values to a K-dimensional vector {\displaystyle \sigma (\mathbf {z} )}\sigma (\mathbf {z} ) of real values in the range [0, 1] that add up to 1. The function is given by

{\displaystyle \sigma (\mathbf {z} )\_{j}={\frac {e^{z\_{j}}}{\sum \_{k=1}^{K}e^{z\_{k}}}}}\sigma (\mathbf {z} )\_{j}={\frac {e^{z\_{j}}}{\sum \_{k=1}^{K}e^{z\_{k}}}} for j = 1, …, K.

The softmax function is used in various multiclass classification methods, such as multinomial logistic regression,[1]:206–209 multiclass linear discriminant analysis, naive Bayes classifiers, and artificial neural networks.[2] Specifically, in multinomial logistic regression and linear discriminant analysis, the input to the function is the result of K distinct linear functions, and the predicted probability for the j'th class given a sample vector x and a weighting vector w[further explanation needed] is:

{\displaystyle P(y=j\mid \mathbf {x} )={\frac {e^{\mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{j}}}{\sum \_{k=1}^{K}e^{\mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{k}}}}}{\displaystyle P(y=j\mid \mathbf {x} )={\frac {e^{\mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{j}}}{\sum \_{k=1}^{K}e^{\mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{k}}}}}

This can be seen as the composition of K linear functions {\displaystyle \mathbf {x} \mapsto \mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{1},\ldots ,\mathbf {x} \mapsto \mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{K}}\mathbf {x} \mapsto \mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{1},\ldots ,\mathbf {x} \mapsto \mathbf {x} ^{\mathsf {T}}\mathbf {w} \_{K} and the softmax function (where {\displaystyle \mathbf {x} ^{\mathsf {T}}\mathbf {w} }\mathbf {x} ^{\mathsf {T}}\mathbf {w} denotes the inner product of {\displaystyle \mathbf {x} }\mathbf {x} and {\displaystyle \mathbf {w} }\mathbf {w}). The operation is equivalent to applying a linear operator defined by {\displaystyle \mathbf {w} }\mathbf {w} to vectors {\displaystyle \mathbf {x} }\mathbf {x}, thus transforming the original, probably highly-dimensional, input to vectors in a K-dimensional space {\displaystyle R^{K}}{\displaystyle R^{K}}.

注：softmax函数的本质就是将一个K

维的任意实数向量压缩（映射）成另一个K维的实数向量，其中向量中的每个元素取值都介于（0，1）之间。

x,w 点积就是上图中Z1，Z2，Z3的计算。将高维的输入x转化一个K维的实数，即K类的各自的概率。

E, <https://tracholar.github.io/wiki/machine-learning/cs231n.html>

四：后记：

理想的分类器应当是除了真实标签的概率为1，其余标签概率均为0，这样计算得到其损失函数为-ln(1)=0. 损失函数越大，说明该分类器在真实标签上的分类概率越小，性能也就越差。当损失函数接近正无穷时表明训练发散，需要调小学习速率。

在ImageNet-1000分类问题中，初始状态为均匀分布，每个类别的分类概率均为0.001，此时损失函数-ln(0.001)=ln(1000)=6.90775.., 当loss总在6.9左右时，说明没有训练收敛的迹象，尝试调大学习速率，或者修改权值初始化方式。

参考：知乎

http://blog.csdn.net/u014422406/article/details/52805924