CN 43-1258/TP
ISSN 1007-130X

计算机工程与科学
Computer Engineering & Science

第 41 卷第 5 期 2019 年 5 月
Vol. 41, No. 5, May. 2019

文章编号:1007-130X(2019)05-0796-08

通过 K-means 算法实现神经网络的加速和压缩*

陈桂林,马　胜,郭　阳,李艺煌,徐　睿
(国防科技大学计算机学院,湖南 长沙 410073)

摘　要:近年来,以神经网络为代表的机器学习领域快速发展,已广泛地应用于语音识别和图像识别等各个工业领域。随着应用环境越来越复杂,精度要求越来越高,网络规模也越来越大。大规模神经网络既是计算密集型又是存储密集型结构,其中卷积层是计算密集型层次,全连接层是存储密集型层次。前者的处理速度跟不上存取速度,后者的存取速度跟不上处理速度。基于神经网络本身训练的预测准确率置信区间,提出了一种使用 K-means 加速和压缩神经网络的方法。通过将卷积过程中的输入特征图采用 K-means 压缩来减少计算量;通过将全连接层的权重压缩来减少存储量。所提方法对 AlexNet 网络单个卷积层的计算量最多能降低 2 个数量级,加入合适的 K-means 层,整个网络的处理时间加速比能达到 2.077,对网络压缩率达到 8.7%。

关键词:神经网络;置信区间;加速;聚类压缩
中图分类号:TP389.1　　**文献标志码**:A
doi:10.3969/j.issn.1007-130X.2019.05.005

Towards convolutional neural network acceleration and compression via K-means algotrithm

CHEN Gui-lin, MA Sheng, GUO Yang, LI Yi-huang, XU Rui
(School of Computer, National University of Defense Technology, Changsha 410073, China)

Abstract: In recent years, the field of machine learning develops rapidly. As a typical representative, neural networks are widely used in various industrial fields, such as speech recognition and image recognition. As the environment of application becomes more complex, the accuracy requirements become higher, and the network scale becomes larger. Large-scale neural networks are both computation-intensive and storage-intensive. The convolutional layer is computation-intensive and the fully connected layer is storage-intensive. The processing speed of the former cannot keep up with its memory access speed, while the access speed of the later cannot keep up with its processing speed. Based on the confidence interval of the prediction accuracy of neural network training, we propose a neural network acceleration and compression method using the K-means algorithm. We reduce the amount of calculation by compressing the input feature map during the convolution process; and reduce the amount of storage by compressing the weight of the fully connected layer. The proposed method can greatly reduce the calculation amount of a single convolution layer of AlexNet network by up to 100 times. By adding appropriate K-means layer, the speedup of the processing time of the whole network can reach 2.077, and the network compression can reach 8.7%.

Key words: neural network; confidence interval; acceleration; cluster compression

* 收稿日期:2018-08-05;修回日期:2018-10-16
基金项目:国家自然科学基金(61672526);国防科技大学科研计划(ZK17-03-06)
通信作者:马胜(masheng@nudt.edu.cn)
通信地址:210007 江苏省南京市秦淮区后标营 18 号国防科技大学第 63 研究所
Address: The 63rd Research Institute, National University of Defense Technology, 18 Houbiaoying, Qinhuai District, Nanjing 210007, Jiangsu, P. R. China

1 引言

从 21 世纪的第 1 个 10 年末开始,随着大数据的兴起和计算能力的不断提升,机器学习的研究开始回暖,其中以人工神经网络为代表的深度学习技术发展很快,已广泛地应用到许多人工智能应用中,包括图像识别[1]、自然语言处理[2]和机器人技术等。以前实现神经网络的平台主要是通用处理器,尤其像图像处理单元 GPU(Graphics Processing Unit)更是 DNN(Deep Neural Network)加速的主流选择。但是,随着神经网络的应用越来越广,网络本身越来越复杂,而大型网络需要增加计算能力和内存需求。通用处理器的灵活性限制了其处理特定神经网络结构时的效率。为了实现更高的可扩展性、性能和能耗,目前有 2 个发展趋势渐渐成为新的主流。

第 1 个趋势是研究专用加速器来加速神经网络的处理。不管是在学术界还是在工业界,近几年来,专用加速器的发展都非常快。如 Google 的 TPU(Tensor Processing Unit)[3],中国科学院寒武纪团队研发的 DianNao 系列[4-7]、麻省理工学院(MIT)研发的 Eyeriss[8]等等。这些加速器的共同特点就是设计专用的计算结构和独特的数据流调度来加速神经网络卷积层和全连接层的处理。这些加速器有一个显著的问题,那就是需要不断地访问片外 DRAM。这是由于在处理大规模神经网络时,片上的 SRAM 不能完全存放网络的预训练权重(比如 VGG(Visual Geometry Group)模型的权重大小为 250 MB 左右)。但是,片外的 DRAM 访问会显著增加能耗。据统计[9,10],片外 DRAM 访问的功耗是片上 SRAM 的 200 倍。

为了解决片外访存的功耗问题,神经网络发展的第 2 个趋势就是压缩。目前常用的压缩方法包括权重量子化[10,11]、修剪连接[12]和使用低秩逼近[13],这些方法都能达到不错的压缩效果。但是,它们也存在一些缺点,比如破坏了网络的结构,需要再训练,增加了训练复杂度等等。

一种理想的网络加速和压缩方案应该是能够保持规则的网络结构,不需要额外地再训练,在保证准确率的前提下达到尽可能高的压缩比和加速比。为了达到这 3 个目的,本文提出了基于 K-means 算法的加速和压缩方案。表 1 为几种常见网络的权重和浮点操作数大小,其中 C-Weights-Rate 表示卷积层权重占比,F-Weights-Rate 表示全连接层权重占比,C-Operations-Rate 表示卷积层浮点操作占比,F-Operations-Rate 表示全连接层浮点操作占比。通过表 1 可以看出,神经网络卷积层的参数很少但是计算量很大,全连接层的参数量很大但是计算量很小。因此,本文通过在卷积层前加入一个 K-means 层来减少卷积层的计算量,通过压缩全连接的权重来达到整个网络压缩的目的。

第 1 步将卷积层的输入特征图聚类,假如使用一个 5×5 的卷积核以步长为 1 的方式去卷积一个 16×16 的输入特征图,得到一个 10×10 的输出特征图。聚类前共需要 2 500 次乘法操作(10×10×5×5)。当使用 K-means 算法将特征图所有数聚成 16 类时,使用这 16 类和所有卷积核的数相乘得到 400 个结果(400 次乘法计算),所有输出特征图的结果将会是这 400 个数中的 25 个数的累加结果。这样就达到了减少计算量的目的。

第 2 步具体实现思想是通过聚类算法将全连接层参数聚类,每个权重值有对应的聚类后的值,片上存放每个权重的聚类索引值和整个全连接层的聚类值。由于索引值通常只有 4～5 位(根据具体聚多少类决定),相较于原先存放所有的 32 位权重,达到了压缩的目的。与 Han 等人[10]提出的压缩方式不同的是,本文基于 Minerva 等人[14]提出的置信区间(网路模型本身由于初始条件不同等因素,训练的网络有一定的预测准确率波动)这一概念,直接针对训练好的权重文件做聚类压缩。通过调整聚类数目,使压缩后的预测准确率在置信区间内波动,这样做既能减少再训练的过程,也能达到压缩的目的。

Table 1 Weight size and floating-point operand of common networks

表 1 常见网络的权重和浮点操作数

网络	Weights /KB	Operations /(FLOPS)	C-Weights-Rate /%	F-Weights-Rate /%	C-Operations-Rate /%	F-Operations-Rate /%
LeNet-300-100	1 070	0.2M	-	100	-	100
LeNet-5	240	2.2M	4.17	95.83	82.4	17.6
Cifar10_quick	600	11.7M	56.92	43.08	99.5	0.5
AlexNet	240	832M	6.01	93.99	93.2	6.8

2 通过 K-means 层实现加速

由于在神经网络的训练阶段,初始化条件不同导致每次训练的权重结果并不完全收敛,那么使用不同的训练结果在测试集上测试会造成预测准确率的小幅度波动,我们将这个网络模型本身结构所带来的波动称作置信区间[14]。只要经加速和压缩后的网络在测试集上的预测准确率波动不超出置信区间范围,则称该方法是有效的。表 2 列出了常见的 4 个网络的置信区间。

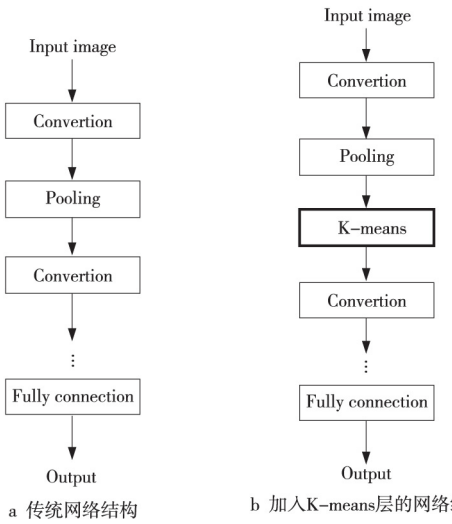
Table 2 Confidence interval of the common networks

表 2 常见网络的置信区间 %

网络	Top_1 Error	Top_2 Error	Confidence Interval
LeNet-300-100	98.33	-	0.30
LeNet-5	99.26	-	0.25
Cifar10_quick	75.75	-	0.34
AlexNet	58.00	80.86	0.81

在过往的研究中,对神经网络的加速优化主要是根据网络的计算特点设计独特的数据流,优化硬件处理流程,尽量减少不必要的运算和访存。比如 TPU[3] 设计的脉动阵列结构,Eyeriss[8] 采用的行固定流结构都是为了减少不必要的访存,CNV (CNVlutin)[15] 的激活 0 值跳过,Cambricon[16] 的权重 0 值跳过都是为了减少不必要的计算。

本文所提的加速方法主要是通过添加一个 K-means 层来减少卷积层的运算。因为在卷积层的运算过程中,权重数据取出后会经过多次重用,取数的周期远远小于处理权重的周期。如果想要提升整个网络的处理速度,就应该从减少卷积层的处理时间着手。卷积层的处理主要是乘累加操作,通过 K-means 层后的输入数据聚成了固定的 k 类,只需要在卷积层中计算这 k 类数据和卷积核的相乘,将其结果做成一个查找表,原先所有输入数据与卷积核相乘的结果都可以通过这个查找表得到,这样就可以减少大量的乘法操作。传统的网络结构如图 1a 所示,加入 K-means 层的网络结构如图 1b 所示。



a 传统网络结构 b 加入K-means层的网络结构

Figure 1 Processing structure of the neural network

图 1 神经网络处理结构

K-means 聚类[17] 的思想并不复杂,整个算法是一个重复移动类中心点的过程,首先从 n 个数据对象中任意选择 k 个对象作为初始聚类中心,通过不断地迭代,将类的中心点移动到所包含成员的平均位置,然后重新划分其内部成员,直到所有成员到所属类的距离最近为止。在加入 K-means 层以后,通过对即将输入卷积层的输入数据聚类,使用聚类的值和所有的卷积核的值相乘,其得到的结果就可以表示卷积核卷积输入特征图所要得出的全部结果。图 2 给出了使用 K-means 后的卷积层处理模式。

如图 2 所示,使用一个 3×3 的卷积核以步长为 1 的方式卷积一幅 5×5 的输入特征图,最终可以得到一幅 3×3 的输出特征图,在 K-means 处理前,需要进行 3×3×5×5 次乘法运算(每个输出像素点都需要 9 次乘法操作)。在使用 K-means 层以后,5×5 的输入特征图被聚类成了 5 个聚类中心。卷积的所有操作都将只是这 5 个数和卷积核的值相乘。我们可以先将所有的相乘结果计算出

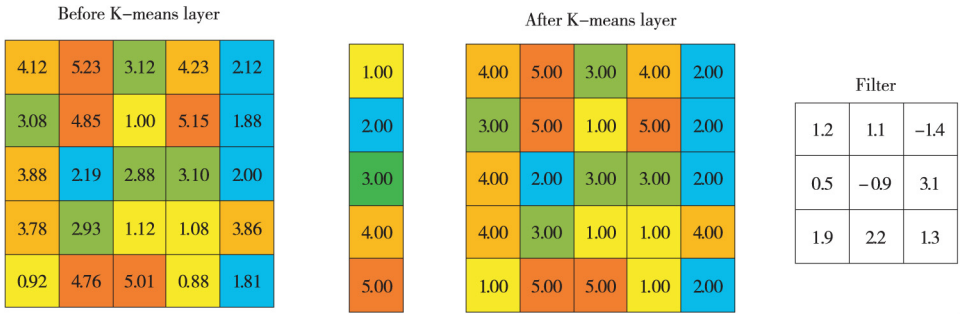


Figure 2 Input data format before and after adding K-means layer

图 2 加入 K-means 层前后的输入数据格式

来,在卷积时直接通过索引取对应数据相乘的结果即可。这种操作需要的乘法计算次数为 5×3×3 次。加速比的计算公式如下所示:

$$\sigma = \frac{T_{\text{mut}} + T_{\text{add}}}{T_{\text{k-mut}} + T_{\text{k-add}} + T'_{\text{mut}} + T'_{\text{add}}}$$

其中,T_{mut} 和 T_{add} 表示不使用 K-means 时卷积层所需的乘法操作和加法操作的时间,$T_{\text{k-mut}}$ 和 $T_{\text{k-add}}$ 表示 K-means 算法所需乘法操作和加法操作的时间,T'_{mut} 和 T'_{add} 表示使用 K-means 算法后卷积层计算所需乘法和加法操作的时间。网络越大,输入特征图越大,所聚类别越少,加速效果越好,这些操作都必须在满足预测准确率的置信区间内进行。

3 K-means 聚类压缩

针对神经网络压缩的方案有很多,文献[11-13]分别介绍了 3 种不同的压缩方法。其中 Data-free pruning[11] 方法减少了 33%的权重参数,但是精度降低很大。它主要是采取修剪相似神经元的做法,在神经网络中只要能找到一个相似权重对,则删掉其中一个,与以往剪枝操作不同的是,以前的剪枝通常是去除单个权重,而 Data-free pruning 的做法是直接删除整个神经元。值得注意的是,该方法是基于训练好的模型,剪枝后不需要访问训练数据进行再训练。Deep Fried Convnets[12] 是作用在全连接层的一种压缩方法,使用自适应快速变换(Adaptive Fast-food transform)替换网络中的全连接层实现。具有自适应变换的卷积神经网络称之为 Deep Fried Convnets,这种网络与标准网络相比,只使用一半的参数就能在 ImageNet 上取得相同的预测性能。SVD 指利用奇异值分解的方法来做张量分解,就是把一个卷积网络参数矩阵通过张量分解,用它的低秩特性做逼近。这种方法能有效减少参数数量但是会带来较大的错误率增长。Denton 等人[13]在 SVD 的基础上利用卷积神经网络的线性特性,使 AlexNet 网络压缩率达到 20%,将精确度损失控制在 0.9%之内。

以上几种方法都存在一些缺点,比如破坏了网络规整结构,增加了训练复杂性,降低了过多的准确率等等,因此本文提出了基于 K-means 的压缩方法来克服这几个缺点。图 3 解释了 K-means 算法是如何应用到神经网络中并达到压缩的目的。与上一节不同的是,该操作是在推导前就完成的,是针对预训练模型的全连接层权重所作的压缩操作,这是由于全连接层的权重数据量大、计算量小(如表 1 所示)。针对其做压缩操作既能达到较高的压缩率又能保持较高的预测准确率。

如图 3 所示,聚类压缩前的权重矩阵存放的是 32 位精度的权重操作数,对其聚类压缩后,将聚类后的值编码,权重矩阵存放每个权重的索引。因为索引值的位数(由聚类的类别决定)远小于操作数的位数,所以通过存放聚类后的索引值,可以达到压缩的目的。压缩比的计算公式为:

$$r = \frac{N_c \times b + N_f \times \log_2 k + k \times b}{N \times b}$$

其中,r 代表压缩率,N 表示整个网络的权重数目,N_c 表示卷积层的权重数目,b 表示权重的精确度(本文取 float32),N_f 表示全连接层的权重数目,k 表示聚类数,$\log_2 k$ 表示聚类后的索引位数。

4 实验方法

实验硬件平台选择 NVIDIA 的 GTX1080,操作系统选择 ubuntu16.04,使用 SystemC 实现模拟器来计算卷积层的加速结果,使用 Caffe 和 Keras 来验证加入 K-means 层以及使用 K-means 压缩全连接权重文件后的网络预测精度损失。

4.1 数据集

本节选择 Mnist[18]、Cifar-10[19] 和 ILSVRC2012[20] 3 个数据集进行实验,其中 Mnist 是手写数字集;Cifar-10 数据集包含 6 万个 32×32 的彩色图片,共分为 10 种类型,由 Alex Krizhevshy、Vi-

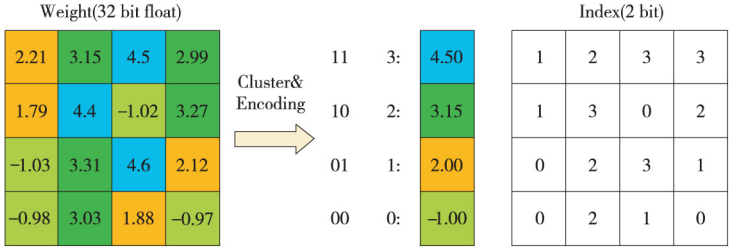


Figure 3 Process of K-means compression

图 3 K-means 压缩的过程

nod Nair 和 Geoffrey Hinton 收集而来,包含 50 000 幅训练图片和 10 000 幅测试图片。ILSVRC2012 是 2012 年的 ILSVRC(ImageNet Large Scale Visual Recognition Challenge)比赛数据,是最流行的一个大规模分类数据,它包含1 000个分类和 100 000+训练数据。

4.2 实验网络

我们选择 AlexNet 来做加速测试,选择表 1 所示的 4 种网络进行聚类压缩的实验,其中 LeNet-300-100 是一个全连接网络,LeNet-5 是最经典的一个卷积神经网络,Cifar10_quick 是 Caffe 开源项目自带的一个网络模型,用于 Cifar-10 数据集的识别,AlexNet 是 2012 年 ILSVRC 大赛的冠军。

4.3 实验细节

做加速测试的方法主要是统计输入数据经过 K-means 层后在卷积层中乘法操作的减少量。我们将上述网络结构每个全连接层进行聚类压缩,比较不同全连接层聚不同类时预测准确率的变化,选取每个层的最优聚类和整个网络的最优聚类。最后再与已有的压缩工作比较压缩率和精确度。

5 结果与分析

5.1 加速的结果

图 4 给出了 AlexNet 每层的参数情况和浮点计算情况。从图 4 中可以看出,第 2 个卷积层和第 4 个卷积层的浮点计算次数最多,我们分别在这 2 个卷积层前加入一个 K-means 层。表 3 给出了当取不同聚类 k 值时整个网络能够取得的加速比。

Table 3 AlexNet's speedup when selecting different k

表 3 选取不同 k 值时 AlexNet 的加速比

k	Before Conv2		Before Conv4	
	σ	Accuracy/%	σ	Accuracy/%
4	1.363	Top1:57.32 Top5:79.68	1.362	Top1:56.96 Top5:80.00
8	1.361	Top1:58.28 Top5:80.52	1.356	Top1:57.60 Top5:79.60
16	1.356	Top1:58.08 Top5:80.64	1.344	**Top1:57.60** **Top5:80.00**
32	1.345	**Top1:58.08** **Top5:80.68**	1.322	Top1:57.24 Top5:79.32
64	1.325	Top1:58.12 Top5:80.60	1.279	Top1:56.80 Top5:79.32

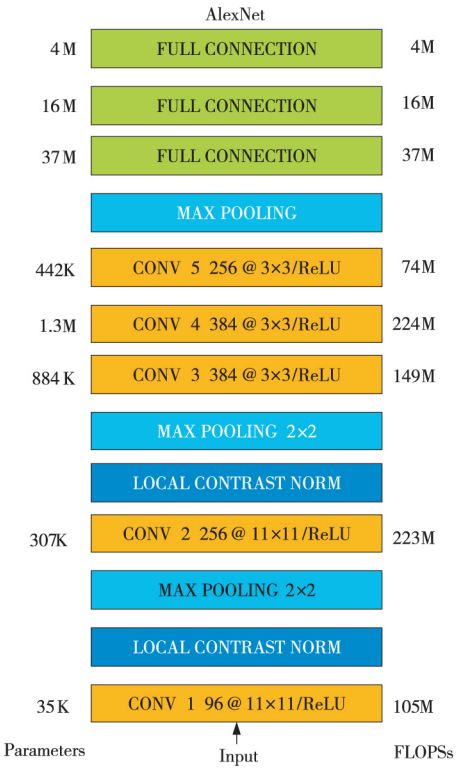


Figure 4 AlexNet's parameters and FLOPS per layer

图 4 AlexNet 每层的参数和浮点计算

通过表 3 的实验结果可知,添加一个 K-means 层能够使网络在不损失预测准确率精度的条件下获得不错的加速比。但是,为了取得更大的加速比,我们考虑同时在 2 个卷积层前都添加 K-means 层,选取合适的 k 值组合,使精度的波动处于置信区间。通过实验得到的 k 值组合是在第 2 个卷积层前选取 k 为 16 的聚类,第 4 个卷积层前选取 k 为 16 的聚类,此时网络的预测准确率处于置信区间(top1:57.36%、top5:79.80%),加速比达到了 2.077。插入 K-means 层是有选择性的,不能所有的卷积层前都加入 K-means 层,这样会大大降低网络的预测准确度,通常选取浮点计算最多的 1 个或 2 个卷积层在其前加入 K-means 层,再选取合适的 k 值以同时满足加速比和预测准确率。

5.2 压缩的结果

表 4~表 7 分别给出了 4 个网络每个全连接层聚不同类后的预测准确率变化。从表中可以看出,对任一网络的任一个全连接层,k(k 表示聚 k 类)从 1 到 2 的变化都会使整个网络预测准确率有一个阶跃式提升。然后 k 从 2 到 8 的变化过程中,预测准确率每次都有小幅度提升。当 $k=8$ 时,聚类压缩后的网络预测准确率已经非常逼近未压缩网络的预测准确率,此后随着 k 的继续增加,网络的预测准确率开始呈现小幅度波动,此时的波动已经处于网络的置信区间内。

Table 4 Prediction accuracy of LeNet-300-100 (98.33%)

表 4 LeNet-300-100 (98.33%)预测准确率 %

全连接层	Weight rate	k						
		1	2	4	8	16	32	64
*Fc*1	88.33	9.74	90.68	96.83	98.35	98.38	98.39	**98.42**
*Fc*2	11.30	9.74	95.95	98.06	98.36	98.43	**98.45**	98.44
*Fc*3	0.37	9.74	97.95	98.24	98.34	98.44	**98.45**	98.44

Table 5 Prediction accuracy of LeNet-5 (99.26%)

表 5 LeNet-5 (99.26%)预测准确率 %

全连接层	Weight rate	k						
		1	2	4	8	16	32	64
*Fc*1	77.98	9.74	97.86	98.86	99.14	**99.26**	99.21	99.24
*Fc*2	16.47	9.74	99.10	99.17	99.21	**99.29**	99.25	99.26
*Fc*3	1.38	9.74	99.07	99.07	99.23	**99.29**	99.24	99.26

Table 6 Prediction accuracy of Cifar10_quick (75.75)

表 6 Cifar10_quick (75.75%)预测准确率 %

全连接层	Weight rate	k						
		1	2	4	8	16	32	64
*Fc*1	42.66	10.14	65.64	71.06	74.68	75.64	**75.84**	75.74
*Fc*2	0.42	10.14	59.88	72.58	**75.80**	75.58	75.74	75.78

Table 7 Prediction accuracy of AlexNet (top_1:58.00% top_5:80.64%)

表 7 AlexNet (top_1:58.00% top_5:80.64%)预测准确率 %

全连接层	Weight rate	k						
		1	2	4	8	16	32	64
*Fc*1	60.52	Top_1:00.12 Top_5:00.52	Top_1:45.44 Top_5:71.60	Top_1:57.28 Top_5:80.46	Top_1:57.96 Top_5:80.36	**Top_1:58.04** **Top_5:80.60**	Top_1:57.88 Top_5:80.48	Top_1:57.68 Top_5:80.76
*Fc*2	26.90	Top_1:00.12 Top_5:00.56	Top_1:13.36 Top_5:27.88	Top_1:57.36 Top_5:80.60	Top_1:57.92 Top_5:80.60	Top_1:57.84 Top_5:80.64	Top_1:57.88 Top_5:80.56	**Top_1:58.04** **Top_5:80.48**
*Fc*3	6.57	Top_1:00.12 Top_5:00.52	Top_1:45.44 Top_5:71.60	Top_1:57.28 Top_5:80.48	Top_1:57.80 Top_5:80.60	Top_1:57.96 Top_5:80.72	**Top_1:58.12** **Top_5:80.56**	Top_1:57.92 Top_5:80.64

通过对实验结果的分析发现,在选取合适 k 值的条件下,可以很好地压缩网络且使它的预测准确率波动不超过网络本身的置信区间。

在选取 k 值时不考虑 $k<4$ 的情况,因为此时的预测准确率波动还不处于网络的置信区间。在 $k\geqslant 4$ 时,直观上希望越靠近输出的层聚越多的类,这样能保证聚类操作对输出的影响较小,对于离输出较远的层我们希望聚更少的类,因为这些全连接层的权重比重大,更小的 k 值能取得更好的压缩效果。但是,最终实验结果表明,在 $k\geqslant 4$ 时,网络的预测准确率并没有随 k 值的继续增大而提高,它实际上呈现的是一个波动趋势。第 1 个全连接层可能选取 $k=32$ 时达到最优,第 2 个全连接层可能选取 $k=16$ 时达到最优。因此,依据距离全连接层的远近选取 k 值大小的说法只能作为参考,实际选取 k 值时,在保证预测准确率在置信区间的前提下,k 值越小越好。

以上只讨论了每个全连接层的局部最优 k 值,对于整个网络来说,并不是所有网络选取每个全连接层最优 k 值进行聚类最终得到的结果也能最优。以 AlexNet 为例,如果按照每个全连接层都选局部最优的原则,*Fc*1(*Fc*1 表示第 1 个全连接层)选 $k=16$,*Fc*2 选 $k=64$,*Fc*3 选取 $k=32$。最终得到的 top_1 准确率 57.60% 和 top_5 准确率 80.56%,这个预测准确率略低于 3 个全连接层都选 $k=64$ 时的 top_1 准确率 57.84%和 top_5 准确率 80.72%。不过这 2 种方案的预测准确率都在置信区间之内。实际在确定一个网络所有全连接

层 k 值时,先考虑采用它的局部最优 k 值组合,如果得到聚类后的网络预测准确率满足置信区间的要求,则使用这些 k 值组合。如果存在更优的 k 值组合方式,能取得更大的压缩比且准确率也不降低(或降低幅度在置信区间内),那么再更新为新的 k 值组合。

表 8 给出了每个网络全连接层取得最大压缩率的 k 值组合,以及它们的预测准确率和压缩率。表 9 给出了 AlexNet 采用聚类压缩和其他方式压缩的压缩比和预测准确率。从 2 个表中可以看出,本文提出的聚类压缩方法相较于其它压缩方法压缩比更大,准确率波动更小。全连接层的权重比重越大,压缩效果越好。

Table 8 Compression rate and accuracy of k-value combinations of the fully connected layer

表 8 全连接层不同 k 值组合的压缩率和准确率

网络	k	Accuracy/%	Compress rate/%
LeNet-300-100	$k1=8, k2=8, k3=8$	98.22	9.12
LeNet-5	$k1=8, k2=8, k3=8$	99.14	8.98
Cifar10_quick	$k1=16, k2=8$	75.56	45.23
AlexNet	$k1=8, k2=8, k3=8$	Top_1:79.44 Top_5:80.04	8.74

Table 9 Compression rate of the compression methods on AlexNet

表 9 各种压缩方法针对 AlexNet 的压缩率

Method	Top_1 /%	Top_5 /%	Compress Rate /%
Baseline Caffemodel[21]	58.00	80.64	100
Data-free pruning[11]	56.60	-	66.6
Fastfood-16-AD[12]	58.07	-	50
Fastfood-32-AD[12]	57.10	-	27
SVD[13]	55.98	79.44	20
K-means Cluster Compress	57.60	80.04	8.74

6 结束语

本文提出了一种基于 K-means 的加速和压缩方法,在部分卷积层前面插入 K-means 层以减少卷积层的浮点运算,通过选取合适的 k 值,能够在几乎不损失精度的情况下使 AlexNet 的推导时间缩短一半。使用 K-means 聚类方法压缩预训练的全连接层权重,与以前的压缩工作相比,它更简单,更容易实现。为了保持预测的准确性,传统的修剪和压缩方法需要大量时间进行再训练。但是,聚类压缩方法只需要对训练后的权重进行聚类,然后选择适当的 k 值来获得更高的压缩比和不变的预测精度。最终,在 AlexNet 上实现了 8.74% 的压缩比。

参考文献:

[1] He K,Zhang X,Ren S,et al. Deep residual learning for image recognition[C]// Proc of 2016 IEEE Conference on Computer Vision and Pattern Recognition,2015:770-778.

[2] Hannun A,Case C,Casper J,et al. Deep speech:Scaling up end-to-end speech recognition[J]. arXiv:1412.5567,2014.

[3] Jouppi N P,Young C,Patil N,et al. In-Datacenter performance analysis of a tensor processing unit[C]// Proc of the 44th International Symposium on Computer Architecture,2017:1-12.

[4] Chen Yun-ji,Chen Tian-shi,Du Zi-dong,et al. DianNao:A small-footprint high-throughput accelerator for ubiquitous machine-learning[C]// Proc of the 19th International Conference on Architectural Support for Programming Languages and Operating Systems,2014:269-284.

[5] Chen Yun-ji,Luo Tao,Liu Shao-li,et al. DaDianNao:A machine-learning supercomputer[C]// Proc of the 47th Annual IEEE/ACM International Symposium on Microarchitecture,2014:609-622.

[6] Liu Dao-fu,Chen Tian-shi,Liu Shao-li,et al. PuDianNao:A polyvalent machine learning accelerator[C]// Proc of the 20th International Conference on Architectural Support for Programming Languages and Operating Systems,2015:369-381.

[7] Du Zi-dong,Fasthuber R,Chen Tian-shi,et al. ShiDianNao:Shifting vision processing closer to the sensor[C]// Proc of the 42nd Annual International Symposium on Computer Architecture,2015:92-104.

[8] Chen Yu-hsin,Emer J,Sze V. Eyeriss:A spatial architecture for energy-efficient dataflow for convolutional neural networks[C]// Proc of the 43rd Annual International Symposium on Computer Architecture,2016:367-379.

[9] Han Song,Pool J,Tran J,et al. Learning both weights and connections for efficient neural network[C]// Proc of Advances in Neural Information Processing Systems,2015:1135-1143.

[10] Han S,Mao H,Dally W J. Deep compression:Compressing deep neural networks with pruning,trained quantization and Huffman coding[J]. Fiber,2015,56(4):3-7.

[11] Srinivas S,Babu R V. Data-free parameter pruning for deep neural networks[J]. arXiv:1507.06149,2015:2830-2838.

[12] Yang Z,Moczulski M,Denil M,et al. Deep Fried convnets[C]// Proc of IEEE International Conference on Computer Vision,2015:1476-1483.

[13] Denton E,Zaremba W,Bruna J,et al. Exploiting linear structure within convolutional networks for efficient evaluation[C]// Proc of the 27th International Conference on Neural Information Processing Systems,2014:1269-1277.

[14] Reagen B,Whatmough P,Adolf R,et al. Minerva:Enabling

low-power, highly-accurate deep neural network accelerators [C] // Proc of ACM/IEEE International Symposium on Computer Architecture, 2016: 267-278.

[15] Albericio J, Judd P, Hetherington T, et al. Cnvlutin: Ineffectual-neuron-free deep neural network computing[C] // Proc of the 43rd International Symposium on Computer Architecture, 2016: 1-13.

[16] Liu Shao-li, Du Zi-dong, Tao Jin-hua, et al. Cambricon: An instruction set architecture for neural networks[C] // Proc of the 43rd International Symposium on Computer Architecture, 2016: 393-405.

[17] Macqueen J. Some methods for classification and analysis of multivariate observations[C] // Proc of Berkeley Symposium on Mathematical Statistics and Probability, 1966: 281-297.

[18] Lecun Y, Cortes C. The Mnist database of handwritten digits[J]. IEEE Signal Processing Magazine, 2012, 29(6): 141-142.

[19] Krizhevsky A, Hinton G. Learning multiple layers of features from tiny images[R]. Toronto: University of Toronto, 2009.

[20] Deng J, Berg A, Satheesh S, et al. ILSVRC-2012[EB/OL]. [2012-12-11]. http://www.image-net.org/challenges/LSVRC/2012/.

[21] Jia Yang-qing. BVLC caffe model zoo[EB/OL]. [2012-12-11]. http://caffe.berkeleyvision.org/model_zoo.

作者简介:

陈桂林(1994-),**男**,**四川绵阳人**,**硕士生**,CCF **会员**(79812G),**研究方向为神经网络加速器**。**E-mail**:cglnudt@163.com

CHEN Gui-lin, born in 1994, MS candidate, CCF member (79812G), his research interest includes neural network accelerator.

马胜(1979-),**男**,**湖南永州人**,**博士**,**副研究员**,**研究方向为计算机体系结构和片上网络**。**E-mail**:masheng@nudt.edu.cn

MA Sheng, born in 1979, PhD, associate research fellow, his research interests include computer architecture, and on-chip networks.

郭阳(1971-),**男**,**浙江东阳人**,**博士**,**研究员**,CCF **会员**(06794S),**研究方向为计算机体系结构**。**E-mail**:guoyang@nudt.edu.cn

GUO Yang, born in 1971, PhD, research fellow, CCF member (06794S), his research interest includes computer architecture.