一种基于动态量化编码的深度神经网络压缩方法

饶 川 [1, 2] 陈靓影 [1, 2] 徐如意 [1, 2] 刘乐元 [1, 2]

摘 要 近年来深度神经网络 (Deep neural network, DNN) 从众多机器学习方法中脱颖而出, 引起了广泛的兴趣和关注. 然而, 在主流的深度神经网络模型中, 其参数数以百万计, 需要消耗大量的计算和存储资源, 难以应用于手机等移动嵌入式设备. 为了解决这一问题, 本文提出了一种基于动态量化编码 (Dynamic quantization coding, DQC) 的深度神经网络压缩方法. 不同于现有的采用静态量化编码 (Static quantitative coding, SQC) 的方法, 本文提出的方法在模型训练过程中同时对量化码本进行更新, 使码本尽可能减小较大权重参数量化引起的误差. 通过大量的对比实验表明, 本文提出的方法优于现有基于静态编码的模型压缩方法.

关键词 深度神经网络, 模型压缩, 动态量化编码, 码本更新

引用格式 饶川, 陈靓影, 徐如意, 刘乐元. 一种基于动态量化编码的深度神经网络压缩方法. 自动化学报, 2019, **45**(10): 1960−1968

DOI 10.16383/j.aas.c180554

A Dynamic Quantization Coding Based Deep Neural Network Compression Method

RAO Chuan[1, 2] CHEN Jing-Ying[1, 2] XU Ru-Yi[1, 2] LIU Le-Yuan[1, 2]

Abstract Recently, deep neural network (DNN) stands out from many machine learning methods and has attracted wide interest and attention. However, it is difficult to apply DNN to mobile embedded devices such as mobile phones due to millions of parameters for the mainstream model of deep neural network, which requires a lot of calculation and storage resources. To address this problem, this paper proposes a deep neural network compressing method based on dynamic quantization coding (DQC). Different from the existing static quantitative coding (SQC) methods, the proposed method updates the quantized codebook in the training process, so as to minimize the error caused by large weight parameters. Numerous experiments show that the proposed method is superior to the existing model compression method based on SQC.

Key words Deep neural network (DNN), model compression, dynamic quantization coding (DQC), codebook update

Citation Rao Chuan, Chen Jing-Ying, Xu Ru-Yi, Liu Le-Yuan. A dynamic quantization coding based deep neural network compression method. *Acta Automatica Sinica*, 2019, **45**(10): 1960−1968

近年来, 深度神经网络在解决机器学习任务时取得了巨大的成功. 2012 年, Krizhevsky 等[1] 首次使用深度神经网络, AlexNet 在 ILSVRC (ImageNet large scale visual recognition competition)[2] 分类任务上获得了冠军, 他们的结果相比传统的机器学习算法在识别精度上提升了近 10 个百分点, 引起学术界和工业界巨大轰动. 从那时起, 各种不同结构的深度神经网络模型如雨后春笋般不断涌现. 2014 年英国牛津大学的 Visual Geometry Group 提出了 VGG[3] 模型, 同时谷歌的研究人员提出了 GoolgLeNet[4], 2015 年 He 等提出了 ResNet[5−6]. 这些模型的网络结构越来越深, 从而能学习到更好的特征以提升模型的性能. 然而, 由于内存和计算能力有限, 随着网络变得越来越深, 对包括移动设备在内的有严格时延要求的有限资源平台而言, 随之增加的模型参数需要消耗更多的计算和存储资源, 难以直接应用于手机等移动嵌入式设备.

虽然将深度神经网络部署到云服务器端, 移动

收稿日期 2018-08-17 录用日期 2019-01-02
Manuscript received August 17, 2018; accepted January 2, 2019
国家重点研发计划 (2018YFB1004504), 中央高校基本业务费 (CCNU19Z02002), 中国博士后科学基金 (2018M632889), 湖北省自然科学基金 (2017CFB504), 湖北省创新研究团队 (2017CFA007), 国家自然科学基金 (61702208, 61807014) 资助
Supported by National Basic Research Program of China (2018YFB1004504), Basic Operating Costs of Central Universities (CCNU19Z02002), China Postdoctoral Science Fund (2018M632889), Hubei Natural Science Foundation (2017CFB504), Foundation for Innovative Research Groups of Hubei Province (2017CFA007), and National Natural Science Foundation of China (61702208, 61807014)
本文责任编委 胡清华
Recommended by Associate Editor HU Qing-Hua
1. 华中师范大学国家数字化学习工程技术研究中心 武汉 430079 2. 华中师范大学教育大数据国家工程实验室 武汉 430079
1. National Engineering Research Center for E-Learning, Central China Normal University, Wuhan 430079 2. National Engineering Laboratory for Big Data for Education, Central China Normal University, Wuhan 430079

端将数据上传服务端处理, 能很好解决上述问题, 但服务器需要耗费昂贵的硬件成本, 导致计算成本过高, 其次移动端在使用的过程对网络要求高, 需要将移动端本地的数据上传到云端服务器进行保存, 在处理一些隐私数据时, 还会存在信息的泄露. 所以将深度学习算法部署到移动端本地非常有意义, 但常用的深度模型具有大量的储存量, 比如 AlexNet 的模型大小量超过 200 MB, VGG 的模型大小量超过 500 MB, 如果将这些网络结构直接运用到手机 APP 中, 用户需要大量的流量下载 APP 文件, 还会占用用户手机的大量内存. 同时, 巨大的模型文件会带来另外一个问题, 手机设备的能源消耗问题, 手机在调用这些文件时会存在大量的能源消耗, 会让手机设备产生大量的热量, 因此深度模型压缩是非常有必要的.

为了解决这一问题, 各种深度模型压缩方法被提出, 这些方法主要有: 模型剪枝、知识精炼 (Knowledge distillation)、低秩分解、精细化的模型结构以及权重共享. 本文主要关注基于权重共享的方法来压缩模型, 权重共享是通过卷积滤波器中相近的权重共享同一量化值从而达到对深度神经网络压缩的目的. 权重共享的方法大致可以分为三类: 聚类编码法、二值/三值量化编码和指数量化编码.

聚类编码法, 指使用聚类的方法, 将权重聚类为若干类, 取每类的聚类中心作为共享的权重. 例如, Gong 等[7] 通过对全连接层参数进行聚类, 取聚类中心作为量化编码, 可以将深度模型 AlexNet 压缩 20 倍, 而 Top-5 准确率仅仅下降 1 %. 类似的, Chen 等[8] 提出了 HashedNets 对全连接层进行量化, 利用哈希函数随机的将权重分到不同的哈希桶 (Hash bucket), 同一哈希桶的权重具有相同的量化值. Han 等[9] 提出了一种深度压缩 (Deep compression) 方法, 通过对卷积层和全连接层的权重进行剪枝, 然后通过 K-Means 聚类进行量化, 最后使用霍夫曼编码 (Huffman coding) 对模型进行保存, 极大的压缩了模型的规模. 但是, 聚类编码法需要大量额外的计算开销用于聚类分析, 算法的训练效率低.

二值/三值量化编码, 将网络的权重、激活值量化为二值或者三值. 例如, Courbariaux 等[10] 提出了一种二值连接 (Binary connect) 方法, 将权重进行二值量化 (量化为 $-1, 1$), 可以将一个深度模型直接压缩 32 倍; 他们随后提出了 Binarized neural networks[11], 将权重和激活值同时进行二值量化, 在压缩网络的同时还可以提高运算效率. Rastegari 等[12] 提出的二值权重网络 (Binary weight networks) 和 XNOR-Net, 在把权重量化为 +1 和 −1 的基础上乘以一个尺度因子, 在大数据集 ImageNet 上取得了不错的压缩效果. Li 等[13] 提出的 HORQ, 相对于 XNOR 只使用到一阶阈值, HORQ 还用到二阶阈值, 让二阶阈值逼近一阶阈值的残差, HORQ 在对准确率影响很小的情况下能将模型缩小 32 倍, 同时加速 30 倍左右. Li 等[14] 提出的 TWN (Ternary weight networks), 在二值网络的基础中引入 0 作为量化权重值. Zhu 等[15] 提出的 TTQ (Trained ternary quantization), 将网络的权重量化为 +1 和 −1 的基础上, 分别乘上两个不同的尺度因子. Cai 等[16] 提出 HWGQ-Net 通过改变网络的激活函数, 将权重量化 1 个比特网络的激活值量化为 2 个比特, 网络只有微小的性能损失. 尽管这类方法将模型中的每一个权重、激活值压缩为一到两位来表示, 但是模型的识别精度也有一定幅度的下降.

指数量化编码, 将权重量化为 2 的 (为整数) 次幂, 以便于嵌入式系统进行二进制移位操作. 该方法由 Zhou 等[17] 首次提出, 利用预训练模型获取固定的指数量化的码本, 然后通过增量量化过程来逐渐量化整个网络. 在使用 5-bit 量化时, 压缩模型的识别率能够保持与原模型一致. 这种方法在量化时, 可以高效率对深度神经网络量化, 但在网络重训练时, 他们提出的增量网络量化方法需要分多个阶段训练, 训练效率较低.

鉴于以上几点, 本文提出一种基于动态量化的深度模型压缩方法, 不同与从预训练模型获取固定码本, 动态量化在训练的过程中也更新码本. 这种量化方式在保证模型性能的同时, 还加快了效率. 同时为了方便嵌入式系统进行移位操作, 本文对所有网络的权值采用指数量化编码, 并通过理论证明得出, 绝对值较大权值参数的量化对模型的精度影响越大. 因此, 本研究通过动态编码使得码本能自适应网络中绝对值较大的权值参数. 本文的主要贡献有以下三点:

1) 提出动态更新码本自适应网络中绝对值较大的权值参数, 尽可能减小这些参数的量化对模型精度的影响;

2) 提出交替迭代算法用于模型训练, 使得权值参数和码本交替更新, 加快训练过程的收敛速度;

3) 大量的对比实验表明, 本文所提的基于动态编码的模型压缩方法总体上优于静态编码的方法.

本文剩余部分的组织结构如下: 第 1 节介绍现有的深度神经网络压缩方法; 第 2 节介绍本文提出的方法, 包括基于权重的动态编码过程以及压缩模型的训练过程; 第 3 节通过大量的对比实验验证本文方法的有效性; 第 4 节总结与展望.

1 相关的工作

本节主要介绍不同的深度神经网络压缩方法. 除了上一节重点介绍的权重共享的方法, 深度神经

网络压缩的方法还包括: 模型的剪枝、知识精炼 (Knowledge distillation)、低秩分解以及精细化的模型结构.

模型的剪枝, 通过评定预训练模型中参数的重要性, 剔除网络中不重要的网络连接, 主要有权重剪枝和滤波器剪枝两种方法. 权重剪枝主要通过判断权重的大小来决定重要性, 一般设定一个阈值进行剔除, 或者根据设定剪切比例进行剔除, 优先将那些权重较小的值剔除例如 Song 等[18] 采用此方法将 AlexNet 模型的参数量减少了 9 倍, VGG 模型的参数量减少 13 倍, 而并没有造成模型精度的下降; Anwar 等[19] 按照一定的比例对每层中的权重随机裁剪, 统计多种随机剪裁下局部最优的作为最终剪裁. 滤波器剪枝则是剔除网络中那些对网络影响较小的卷积滤波器, Li 等[20] 通过对卷积滤波器所有元素绝对值求和, 剔除那些求和较小的滤波器, 从而实现对模型的剪枝; Luo 等[21] 对卷积滤波器剪切前和剪切后激活值的损失进行评定, 剪切那些对损失影响不大的滤波器; Hu 等[22] 通过判定卷积录波器中权重为 0 的参数量来评定剪切标准; Luo 等[23] 提出了一种基于熵值的裁剪方式, 通过特征层的概率分布来评价卷积滤波器的重要性; Yang 等[24] 通过每层需要消耗的能量来进行裁剪, 优先修剪那些消耗大的层. 相对而言, 滤波器剪枝会产生规则的稀疏矩阵, 而权重剪枝则会产生大量不规则的稀疏矩阵, 因此, 在加速方面滤波器剪枝更加有效.

知识精炼, 利用大模型指导小模型, 从而让小模型学到大模型相似的函数映射. Hinton 等[25] 利用训练好的复杂模型指导小模型的训练, 小模型通过优化复杂模型的输出交叉熵和自身的交叉熵, 在模型性能和训练速度上均有所提高. Romero 等[26] 提出的 FitNets 通过添加网络模型中间层的特征作为监督信号, 有效解决由于网络层数过深造成的学习困难. Zagoruyko 等[27] 同样采用大模型的中间特征作对小模型进行监督学习, 让小模型同时学到低、中、高层三个层次的特征输出.

低秩分解, 将原来的矩阵分解成若干个小矩阵, 对分解的小矩阵进行优化调整. Zhang 等[28] 将卷积矩阵变换为二维的矩阵, 结合 SVD 分解, 将 VGG-16 模型加速 4 倍而精度只有微小的下降. Lebedev 等[29] 使用 CP 分解的方法, 将每层网络分解成若干个低复杂度的网络层, 将 AlexNet 的第二个卷积层的速度提升了 4 倍却只增加 1 % 的分类误差.

精细化的模型结构, 通过使用小的卷积单元或者改变卷积方式对模型进行压缩和加速. Iandola 等[30] 提出的 SqueezeNet 使用 1×1 卷积核对上层特征进行卷积降维, 然后使用 1×1 和 3×3 卷积进行特征堆叠, 大大减小了卷积的参数数量. Howard 等[31] 提出的 MobileNets 对每个通道的特征单独卷积, 之后再使用 1×1 卷积对不同通道特征进行拼接. Zhang 等[32] 提出的 ShuffleNet 则是对多通道特征先进行分组后再执行卷积, 避免信息流不通畅的问题. 这些轻量化的模型设计, 极大了减小了模型的参数量和计算量.

这些方法在对深度神经网络的压缩, 使得网络的性能在一定程度上有所下降, 有些压缩算法实现步骤繁琐, 甚至有些方法还对原始的网络结构进行了改变. 而与这些方法相比, 权重共享的方法只对深度神经网络中的权重进行量化, 实现简单, 不会改变模型的网络结构, 本文对深度神经网络的压缩采用了权重共享的方式.

2 动态量化编码的深度神经网络

本文提出的方法由两部分组成: 权重量化与动态编码, 以及基于动态编码的量化模型训练, 本节将详述这两部分内容.

2.1 权重量化与动态编码

为了方便嵌入式系统进行移位运算, 本文采用类似文献 [15] 中的方法, 采用 2 的 n 次幂的形式对神经网络中的权值进行量化, 即当权重量化为 b 比特时, 码本最多有 2^b 个取值. 码本可以表示为:

$$P_l = \{\pm 2^n\}, n \in [n_1, n_2], n \in \mathbf{Z} \tag{1}$$

式中, l 代表深度神经网络的第 l 层, n_1 和 n_2 是两个整数, 满足 $n_1 < n_2$. 由于 n_1 和 n_2 之间有 $n_2 - n_1 + 1$ 个整数, 且码本中正负整数的个数是相等的, 因此码本中总的取值有 $2 \times (n_2 - n_1 + 1) = 2^b$ 个, 即:

$$n_2 - n_1 + 1 = 2^{b-1} \tag{2}$$

亦可引入 0 作为量化值对权重进行编码, 具体形式为:

$$P_l = \{\pm 2^n, 0\}, n \in [n_1, n_2], n \in \mathbf{Z} \tag{3}$$

由于 0 无法表示成 2 的 n (n 为整数) 次幂, 需要额外的一个比特来表示 0 这个量化值. 当 n_1 和 n_2 保持不变时, 式 (3) 需要 $b+1$ 比特来量化权重. 即:

$$n_2 - n_1 + 1 = 2^b \tag{4}$$

虽然将 0 作为量化值引入码本需要增加一个比特, 但是会让网络中产生大量的稀疏矩阵, 有利于网络的正则化, 能在一定程度上抑制过拟合.

无论是否引入 0 进行编码, 当量化的位数确定时, 只要确定 n_1 或 n_2 中任意一个的值, 根据式 (2) 或式 (4) 求得另外一个参数, 从而根据式 (1) 或式

(3) 得到码本. 假设给定预训练的模型, 并将此模型中的网络权值量化为 2 的 n 次幂. 可采用以下式 (5) 使量化误差最小化:

$$\begin{aligned}&\min_{n} err = \min_{n}(|w^+ - 2^n|), w^+ \in |W_l| \\ &\text{s.t.} \quad n \in [n_1, n_2], n \in Z\end{aligned} \tag{5}$$

式中, err 为量化误差, W_l 为预训练模型的第 l 层权重, w^+ 为 $|W_l|$ 中的任一正值权重. 假设 $|W_l|$ 中的两个值 w_1^+ 和 w_2^+, $w_1^+ < w_2^+$, 同时近似满足 $2^q w_1^+ \approx w_2^+$, q 为正整数. w_1^+ 和 w_2^+ 分别量化为 2^{m_1} 和 2^{m_2}, 2^{m_1} 和 2^{m_2} 应满足 $2^q 2^{m_1} = 2^{m_2}$, 则 w_1^+ 和 w_2^+ 的量化误差 err_1 和 err_2 满足以下关系:

$$\begin{aligned}err_1 =& |w_1^+ - 2^{m_1}| \to \Delta \approx 0 \\ err_2 =& \left|w_2^+ - 2^{m_2}\right| = 2^q \times \left|\frac{1}{2^q} w_2^+ - 2^{m_1}\right| \approx \\ & 2^q \times |w_1^+ - 2^{m_1}| = 2^q err_1 \geq err_1\end{aligned} \tag{6}$$

由于量化模型的权值调整大小是由反向传播的梯度和学习率的乘积来决定, 而这两个量都非常的小, 因此 $2^q w_1^+ \approx w_2^+$ 这一假设是容易满足的, 从而可以根据式 (6) 得出当模型进行指数量化时, 量化权值的绝对值越大, 量化误差也越大.

基于这一结论, 我们在量化的过程中优先量化那些权重绝对值较大的值, 即根据 $|W_l|$ 中的最大值确定上限 n_2 的取值. 具体的计算过程如下:

$$n_2 = \mathbf{floor}(\log_2(\max(|W_l|))) \tag{7}$$

其中, floor$(\cdot)$ 表示的是向下取整操作, max$(\cdot)$ 表示的取最大值操作. 确定了 n_2, 就可以通过式 (2) 或式 (4) 得到下限 n_1 的取值, 从而确定码本 P_l.

对于式 (1) 定义的码本, 每个网络权值用码本中最近的量化值进行编码, 具体量化规则如图 1 所示.

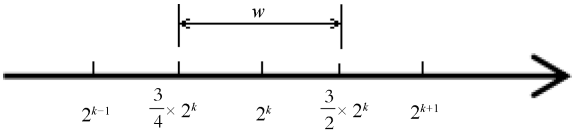


图 1　网络权值的量化规则

Fig. 1　Quantization rules for network weights

可用如下的公式来计算:

$$\hat{w} = \begin{cases} 2^{n_2} \times I(w), & |w| \geq \dfrac{3}{2} \times 2^{n_2} \\ 2^k \times I(w), & \dfrac{3}{4} \times 2^k \leq |w| < \dfrac{3}{2} \times 2^k, \\ & k \in [n_1, n_2], k \in \mathbf{Z} \\ 2^{n_1} \times I(w), & |w| \leq \dfrac{3}{4} \times 2^{n_1} \end{cases} \tag{8}$$

式中, $\hat{w}$ 为量化后的权重, 2^k 为权重 w 的绝对值 $|w|$ 的量化值; $I(w)$ 为指示函数, 用于区分网络中的正负权值. 当 w 大于 0 时, $I(w)$ 为 1; w 为小于或等于 0 时, $I(w)$ 为 -1.

对于式 (3) 定义的码本, 只需要根据码本下限进行截断取 0 值即可, 对应的量化公式如下:

$$\hat{w} = \begin{cases} 2^{n_2} \times I(w), & |w| \geq \dfrac{3}{2} \times 2^{n_2} \\ 2^k \times I(w), & \dfrac{3}{4} \times 2^k \leq |w| < \dfrac{3}{2} \times 2^k, \\ & k \in [n_1, n_2], k \in \mathbf{Z} \\ 0, & |w| < \dfrac{3}{4} \times 2^{n_1} \end{cases} \tag{9}$$

2.2　基于动态编码的量化模型训练

本节主要介绍量化模型的训练过程. 对于初始的深度神经网络模型, 首先基于式 (7) 根据网络中的最大权值确定码本, 然后根据码本对网络权值采用式 (8) 或式 (9) 进行量化, 量化的深度神经网络通过前向传播过程计算网络的损失. 再根据网络损失通过反向传播过程, 对网络权值进行更新. 网络权值的更新会破坏原有的量化, 因此有必要对码本进行更新, 然后使用新的码本对网络权值进行再一次的量化. 整个训练过程码本和权重交替迭代更新, 直到网络损失收敛为止. 在整个迭代过程中, 码本根据网络权值的变化进行动态的更新, 码本的更新和深度神经网络的训练交替进行, 训练流程如图 2 所示. 以下介绍量化模型训练的具体实现细节.

图 2　动态量化编码压缩方法的训练流程

Fig. 2　The process of dynamic quantization coding

对于普通的深度神经网络, 其训练由两个基本过程构成, 即前向传播过程和反向传播过程. 在前向传播过程中, 分层网络的前一层的输出作为后一层的输入, 直到传入网络的最后一层得到整个深度神经网络的输出. 根据输出和标签之间的差异计算损失函数, 其中损失函数的定义如下:

$$\begin{aligned}&\min_{w_l} E(\hat{w}_l) = L(\hat{w}_l) + \lambda R(\hat{w}_l) \\ &\text{s.t.} \quad \hat{w}_l \in P_l, 1 \leq l \leq L\end{aligned} \tag{10}$$

式中, $E(\hat{w}_l)$ 是网络的损失, $R(\hat{w}_l)$ 是正则项, 本文采用 L_2 正则项, λ 是正则项的权值系数. 在反向传播过程中, 网络的残差由后一层逐层向前传递, 网络的权重根据残差计算的梯度进行更新:

$$w_l^{k+1} \leftarrow w_l^k - \gamma \frac{\partial E}{\partial w_l^k} \tag{11}$$

式中, w_l^{k+1} 是更新后的权重, γ 是学习率. 但是, 对于量化模型来说, 对式 (8) 或式 (9) 中指示函数 $I(w)$ 进行求导会导致得到的梯度为 0, 无法更新参数. 需采用 STE (Straight-through estimator) 方法[33] 来处理:

$$\begin{aligned} &\text{前向过程: } q \sim \text{Bernoulli}(p) \\ &\text{反向过程: } \frac{\partial E}{\partial p} = \frac{\partial E}{\partial q} \end{aligned} \tag{12}$$

其中, q 是二项分布函数, $p \in [0,1]$ 的概率. 根据 STE 的处理方法, 在反向求导的过程中, 我们可以这样处理模型中的权重 w_l^k:

$$\frac{\partial E}{\partial w_l^k} = \frac{\partial E}{\partial \hat{w}_l^k} \tag{13}$$

因此在进行反向传播过程时, 式 (11) 可以写成:

$$w_l^{k+1} \leftarrow w_l^k - \gamma \frac{\partial E}{\partial \hat{w}_l^k} \tag{14}$$

采用动态量化编码的方式对深度神经网络压缩, 使得网络的权重在一个动态更新的范围量化. 与静态量化的码表相比, 动态量化后的权重与模型最新更新的权重之间误差更小. 另外, 由于动态更新的码本在训练过程会随着网络的训练误差而间接更新, 所以动态更新码本的方法无需一个预训练的模型作为初始化也能最终使模型收敛.

3 实验与分析

为了验证本文方法的有效性, 我们在标准数据集 MNIST[34]、CIFAR-10[35] 上进行了实验. 其中, MNIST 数据集是一个手写字符数据集, 大小为 28×28 的单通道图像, 包含训练集 60 000 张, 测试集为 10 000 张; CIFAR-10 是一个图像分类数据集, 所有的图像都是大小为 32×32 的三通道彩色图像, 包含 60 000 张图片, 其中训练集为 50 000, 验证集为 10 000.

3.1 MNIST 实验设置

在 MNIST 数据集上, 先使用 LeNet[35] 在不同的损失函数下训练全精度 32 位的模型. 在压缩过程中, 使用预训练的全精度模型作为压缩模型的初始化. 使用的三种损失函数为 Softmax-loss、Softmax-loss 加上 L1 正则项、Softmax-loss 加上 L2 正则项, 分别对应码本中有无 0 两种情况, 实验过程中正则项系数为 0.001, 具体实验结果如下:

通过表 1 ~ 表 3 可以看到, 无论在码本中是否引入 0, 本文的方法均能有效地对网络进行压缩. 同时还可以看到, 在损失函数中引入 L2 正则项有比较好的结果, 因此在后续的实验中只使用 Softmax-loss 加上 L2 正则项作为损失函数.

表 1 LeNet 在 Softmax-loss 下量化效果

Table 1 Quantization performance of LeNet under Softmax-loss

位宽	码本无 0	码本有 0
3	99.29 %	99.22 %
4	99.30 %	99.25 %
5	99.35 %	99.32 %

表 2 LeNet 在 Softmax-loss+L1 下量化效果

Table 2 Quantization performance of LeNet under Softmax-loss and L1

位宽	码本无 0	码本有 0
3	98.69 %	99.25 %
4	99.09 %	99.25 %
5	99.14 %	99.27 %

表 3 LeNet 在 Softmax-loss+L2 下量化效果

Table 3 Quantization performance of LeNet under Softmax-loss and L2

位宽	码本无 0	码本有 0
3	99.26 %	99.29 %
4	99.29 %	99.28 %
5	99.36 %	99.28 %

3.2 CIFAR-10 实验设置

为了清晰地看到压缩前和压缩后的变化, 我们先使用不同深度的 ResNet 训练了全精度 32 位的模型. 在压缩过程中, 为了尽量避免初始化不同对最终实验结果的影响, 以及加快量化模型的训练收敛速度, 均使用预训练好的 32 位模型作为量化模型的初始化. 在预训练和量化压缩过程中, 数据预处理都使用了数据增强的方法, 在原 32×32 的图像边界上填补 0 扩充为 36×36 的图像, 再随机的裁剪为 32×32 的图像, 然后随机左右翻转. 在训练过程中, 都迭代了 80 000 轮, 每轮送进网络一个批次的数据是 128, 初始的学习率为 0.1, 当训练达到 40 000 次学习率为 0.01, 达到 60 000 次之后学习率为 0.001, 训练中使用正则项, 其权值系数设置为 0.001.

3.2.1 对比不同码本的性能

本文在第 3.1 节引入了两种码本, 在量化同样的位数下, 一种在码本中引入了 0 另外一种没有. 将

0 作为量化值引入码表, 会使滤波器产生大量的稀疏矩阵, 这会在一定程度抑制过拟合. 但由于 0 不能表示为 2 的 n (为整数) 次幂这种形式, 需要额外的一个比特来表示, 会影响码表的丰富性. 为了说明这两种量化的差别, 我们做了如下实验:

表 4 ResNet-20 在不同码本下量化效果

Table 4 Quantization performance of ResNet-20 under different codebook

位宽	码本无 0	码本有 0
3	90.07 %	90.78 %
4	91.71 %	91.91 %
5	92.63 %	92.82 %

表 5 ResNet-32 在不同码本下量化效果

Table 5 Quantization performance of ResNet-32 under different codebook

位宽	码本无 0	码本有 0
3	91.44 %	92.11 %
4	92.53 %	92.36 %
5	92.87 %	92.33 %

表 6 ResNet-44 在不同码本下量化效果

Table 6 Quantization performance of ResNet-44 under different codebook

位宽	码本无 0	码本有 0
3	92.68 %	92.53 %
4	93.14 %	93.37 %
5	93.28 %	93.14 %

表 7 ResNet-56 在不同码本下量化效果

Table 7 Quantization performance of ResNet-56 under different codebook

位宽	码本无 0	码本有 0
3	92.72 %	92.69 %
4	93.54 %	93.39 %
5	93.21 %	93.24 %

从表 4 ∼ 7 中可以看到, 两种量化方式均能有效压缩深度神经网络. 当量化位数一定时, 网络越深量化效果越好: 当网络深度一定时, 量化位数越大量化效果越好. 特别在量化位数较大且网络较深时, 采用这种动态量化编码的方法, 甚至可以提升网络的性能.

3.2.2 对比静态量化编码 (SQC) 方法

本文使用的是动态的码本 (Static quantitative coding, DQC), 每次迭代都会对码本进行更新. 为了说明 DQC 的有效性, 我们比较了基于 SQC 和 DQC 的模型性能. 其中, SQC 方法与 DQC 方法不同的地方在于量化模型的训练过程中不对码本进行更新. 基于 SQC 方法的实验结果如表 8, 通过表 8 可以看到深度模型的网络结构越深, 量化的位数越大, 量化效果越好.

为了更加清楚地显示 SQC 和 DQC 两种方法训练得到模型性能的差异, 我们将静态码表的结果减去动态码表的结果, 具体结果如图 3 和图 4 所示.

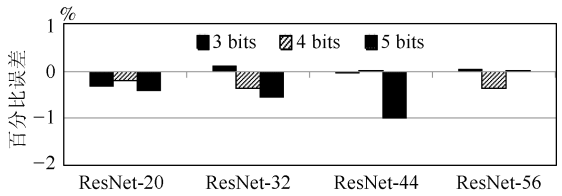


图 3 码本中无 0, SQC 和 DQC 的量化比较

Fig. 3 Quantization performance of SQC and DQC with 0 in codebook

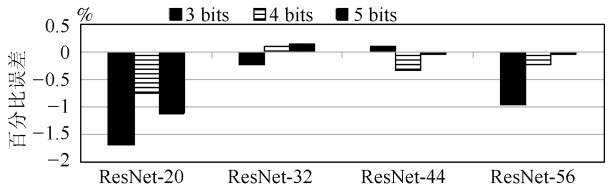


图 4 码本中有 0, SQC 和 DQC 的量化效果比较

Fig. 4 Quantization performance of SQC and DQC without 0 in codebook

从图 3 和图 4 中可以看到, 除了个别情况, 绝大数情况动态变化的码本比固定码本对深度神经网络的压缩效果更好, 特别是在网络较浅时, 动态量化编码的效果比静态量化的效果更加明显. 在使用动态更新的码本方法时, 码本随着权重的更新而更新, 深度神经网络的权重在新的码本内量化, 这会在一定程度上减小由于量化造成的误差; 而固定的码本只与预训练的权重有关系, 量化得到的模型权重与预训练模型中权重误差较小. 显然, 采用动态量化编码的方式, 能更好的减小由于量化造成的影响.

3.3 对比现有方法

本文与 Deep compression[9] 做了对比, 由于文

表 8 固定码本下量化效果

Table 8 Quantization performance of SQC

网络	3 bits 码本 SQC 无 0	3bits 码本 SQC 有 0	3 bits 码本 SQC 无 0	3 bits 码本 SQC 有 0	3 bits 码本 SQC 无 0	3 bits 码本 SQC 有 0
ResNet-20	92.72 %	92.69 %	92.72 %	92.69 %	92.72 %	92.69 %
ResNet-32	93.54 %	93.39 %	92.72 %	92.69 %	92.72 %	92.69 %
ResNet-44	93.21 %	93.24 %	92.72 %	92.69 %	92.72 %	92.69 %
ResNet-56	93.21 %	93.24 %	92.72 %	92.69 %	92.72 %	92.69 %

献 [9] 是对网络进行了剪枝之后再进行量化. 为了对比的公平性, 我们在此处做了和文献 [9] 同样的处理, 使用的数据集是 MNIST, 将 LeNet 第一个卷积层 66 % 的小权重置为 0, 第二个卷积层 12 % 较小权重置为 0, 第一个全连接层 8 % 的小权重置为 0, 第二个全连接层 19 % 的小权重置为 0. 对于这些置为 0 的权重, 在更新过程中不进行求导运算. 此处本文的方法码本中没有引入 0, 具体实验结果如下:

从表 9 可以看到, 在相同条件下, 我们的方法有一定的优势.

为了进一步说明我们的方法, 本文与 INQ[15] 的结果做了比较. 这里主为了实验的客观性, 只与 INQ 做了对比实验, 由于本文的量化方法与 INQ 相近, 都是将权重量化为 2 的 n (n 为整数) 次幂这种形式, 从而在对比实验时避免编码形式的影响. 我们使用相同的数据、初始网络结构, 压缩到同样的位数, INQ 使用 4 个步骤进行量化, 每次量化比例: 0.50, 0.75, 0.85, 1.00, 两种方法均量化 5 bits, 结果如下:

表 9 Deep compression 与 DQC 的实验比较

Table 9 Comparison of deep compression and DQC

压缩方法	位宽	准确率
Deep compression	5	99.20 %
DQC	5	99.70 %

表 10 量化为 5 bits 时 INQ 和 DQC 在 CIFAR-10 上的准确率比较

Table 10 Compare the accuracy of INQ and DQC on CIFAR-10 with 5 bits

网络	INQ	DQC 码本无 0	DQC 码本有 0
ResNet-20	91.01 %	92.63 %	92.82 %
ResNet-32	91.78 %	92.87 %	92.33 %
ResNet-44	92.30 %	93.28 %	93.14 %
ResNet-56	92.29 %	93.21 %	93.24 %

从表 10 可知, 无论网络的层数多深, 码本中是否引入 0, 使用动态量化编码的结果均优于 INQ 的方法, 进一步说明了我们方法的有效性.

4 总结与展望

本文提出了一种基于动态量化编码的深度神经网络压缩方法. 为了方便在嵌入式系统采用移位操作, 本文对网络中的权值采用指数量化编码, 通过理论推导得出, 将模型量化为指数形式时, 绝对值较大权值参数的量化对模型引起的误差也越大. 为此, 本文采用动态量化编码, 在反向传播更新网络权值后, 对码本进行更新以自适应模型中的绝对值较大的权值参数, 减小这些参数的量化对模型精度的影响. 本文还讨论了静态和动态两种不同码本进行编码时压缩模型的性能. 通过实验表明, 深度神经网络越深, 压缩位数越大, 压缩效果越好; 动态量化编码的方法优于静态量化的方法; 本文方法在网络压缩 10.67 倍时准确率还有提升. 虽然本文为了说明动态量化的优越性能, 使用不同深度和量化位宽对深度神经网络压缩进行了大量实验, 但目前只对小数据集进行实验, 后续将在更大的数据集上进行实验.

References

1 Krizhevsky A, Sutskever I, Hinton G E. ImageNet classification with deep convolutional neural networks. In: Proceedings of International Conference on Neural Information Processing Systems. Vancouver, Canada: Curran Associates Inc, 2012. 1097−1105

2 Russakovsky O, Deng J, Su H, et al. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 2015, **115**(3): 211−252

3 Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *Computer Science*, 2014

4 Szegedy C, Liu W, Jia Y Q, et al. Going deeper with convolutions. In: Proceeding of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, USA: IEEE, 2015. 1−9

5 He K, Zhang X, Ren S, et al. Deep Residual Learning for Image Recognition. IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA: IEEE, 2016. 770−778

6 He K, Zhang X, Ren S, et al. Identity Mappings in Deep Residual Networks. In: Proceeding of the European Conference on Computer Vision. Springer International Publishing, 2016. 630−645

7 Gong Y C, Liu L, Yang M, Bourdev L. Compressing deep concolutional net-works using vector quantization. arXiv preprint, arXiv: 1412.6115v1, 2014.

8 Chen W, Wilson J T, Tyree S, et al. Compressing Neural Networks with the Hashing Trick. *Computer Science*, 2015: 2285−2294

9 Han S, Mao H, Dally W J. Deep compression: compressing deep neural networks with pruning, trained quantization and Huffman coding. *Fiber*, 2015, **56**(4): 3−7

10 Courbariaux M, Bengio Y, David J P. BinaryConnect: training deep neural networks with binary weights during propagations. arXiv preprint, arXiv: 1511.00363, 2015.

11 Courbariaux M, Hubara I, Soudry D, et al. Binarized neural networks: training deep neural networks with weights and activations constrained to +1 or −1. arXiv preprint, arXiv: 1602.02830, 2016.

12 Rastegari M, Ordonez V, Redmon J, et al. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In: Proceedings of the European Conference on Computer Vision. Springer, Cham, 2016. 525−542

13 Li Z, Ni B, Zhang W, Yang X, Gao W. Performance Guaranteed Network Acceleration via High-Order Residual Quantization. In: Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, 2017. 2603−2611

14 Li F, Zhang B, Liu B. Ternary weight networks. arXiv preprint, arXiv: 1605.04711, 2016.

15 Zhu C Z, Han S, Mao H Z, Dally W J. Trained ternary quantization. arXiv preprint, arXiv: 1612.01064, 2016.

16 Cai Z, He X, Sun J, Vasconcelos N. Deep Learning with Low Precision by Half-Wave Gaussian Quantization. IEEE Conference on Computer Vision and Pattern Recognition (CVPR). arXiv preprint, arXiv: 1702.00953, 2017.

17 Zhou A J, Yao A B, Guo Y W, Xu L, Chen Y R. Incremental network quantization: towards lossless CNNs with low-precision weights. arXiv preprint, arXiv: 1702.03044, 2017.

18 Song Han, Jeff Pool, John Tran, William J. Dally. Learning Both Weights and Connections for Efficient Neural Networks. arXiv: 1506.02626, 2015.

19 Anwar S, Sung W Y. Coarse pruning of convolutional neural networks with random masks. In: Proceedings of the Int′l Conference on Learning and Representation (ICLR). IEEE, 2017. 134−145

20 Li H, Kadav A, Durdanovic I, Samet H, Graf H P. Pruning filters for efficient ConvNets. In: Proceedings of the Int′l Conference on Learning and Representation (ICLR). IEEE, 2017. 34−42

21 Luo J H, Wu J, Lin W. ThiNet: A Filter Level Pruning Method for Deep Neural Network Compression. 2017: arXiv: 1707.06342

22 Hu H, Peng R, Tai Y W, Tang C K. Network trimming: a data-driven neuron pruning approach towards efficient deep architectures. In: Proceedings of the Int′l Conference on Learning and Representation (ICLR). IEEE, 2017. 214−222

23 Luo J, Wu J. An Entropy-based Pruning Method for CNN Compression. CoRR, 2017. abs/1706.05791

24 Yang T, Chen Y, Sze V. Designing Energy-Efficient Convolutional Neural Networks Using Energy-Aware Pruning. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 6071−6079

25 Hinton G, Vinyals O, Dean J. Distilling the Knowledge in a Neural Network. *Computer Science*, 2015, **14**(7): 38−39

26 Romero A, Ballas N, Kahou SE, Chassang A, Gatta C, Bengio Y. Fitnets: Hints for thin deep nets. In: Proceedings of the Int′l Conference on Learning and Representation (ICLR). IEEE, 2017. 124−133

27 Zagoruyko S, Komodakis N.Paying More Attention to Attention: Improving the Performance of Convolutional Neural Networks via Attention Transfer. CoRR, 2016. abs/1612.03928

28 Zhang X, Zou J, He K, et al. Accelerating very deep convolutional networks for classification and detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2016, **38**(10): 1943−1955

29 Lebedev V, Ganin Y, Rakhuba M, et al. Speeding-up convolutional neural networks using fine-tuned CP-decomposition. *Computer Science*, 2015.

30 Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and < 0.5 MB model size. 2016.

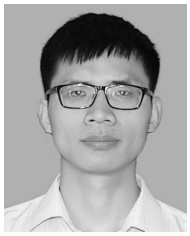
31 Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. 2017. arXiv: 1704.04861

32 Zhang X, Zhou X, Lin M, et al. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. 2017.

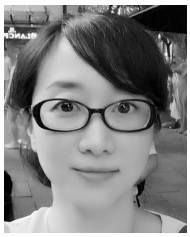
33 Bengio, Yoshua, Léonard, Nicholas, and Courville, Aaron. Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv: 1308.3432, 2013.

34 LeCun, Bottou, Bengio,Haffner. Gradient-based learning applied to document recognition. In: Proceedings of the IEEE, 1998. 86(11): 2278−2324

35 Krizhevsky A. Learning multiple layers of features from tiny images. *Handbook of Systemic Autoimmune Diseases*, 2009, **1**(4).



饶 川 华中师范大学国家数字化学习工程技术研究中心硕士研究生. 2015 年获得湖北大学计算机与信息工程学院学士学位. 主要研究方向为深度模型的压缩与加速.
E-mail: raoguoc@163.com
(**RAO Chuan** Master student at National Engineering Research Center for E-Learning, Central China Normal University. He received his bachelor degree from Hubei University in 2015. His research interest covers deep neural network compression and acceleration.)



陈靓影 华中师范大学国家数字化学习工程技术研究中心教授. 2001 年获得南洋理工计算机科学与工程系博士学位. 主要研究方向为图像处理, 计算机视觉, 模式识别, 多媒体应用. 本文通信作者.
E-mail: chenjy@mail.ccnu.edu.cn
(**CHEN Jing-Ying** Professor at National Engineering Research Center for E-Learning, Central China Normal University. She received her Ph. D. degree from the School of Computer Engineering, Nanyang Technological University, Singapore in 2001. Her research interest covers image processing, computer vision, pattern recognition, and multimedia applications. Corresponding author of this paper.)

徐如意 华中师范大学国家数字化学习工程技术研究中心算法工程师. 2008 年获得武汉科技大学学士学位, 2016 年获得华中科技大学硕士学位. 主要研究方向为计算机视觉及多媒体应用.
E-mail: 86798653@qq.com
(**XU Ru-Yi** Algorithmic engineer at the National Engineering Research Center for E-Learning, Central China Normal University. He received his bachelor degree from Wuhan University of Science and Technology, and master degree from Huazhong University of Science and Technology in 2008 and 2016 respectively. His research interest covers computer vision and multimedia applications.)

刘乐元 华中师范大学国家数字化学习工程技术研究中心副教授. 主要研究方向为计算机视觉, 模式识别, 多模态人机交互. E-mail: lyliu@mail.ccnu.edu.cn
(**LIU Le-Yuan** Associate professor at the National Engineering Research Center for E-Learning, Central China Normal University. His research interest covers computer vision, pattern recognition, and multimodal human-computer interaction.)