上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

学士学位论文

BACHELOR'S THESIS

论文题目：混合比特量化网络建模及准确率提升研究

学生姓名：罗　　钦

学生学号：515021910326

专　　业：测控技术与仪器

指导教师：毛　义　梅

学院(系)：电子信息与电气工程学院

混合比特量化网络建模及准确率提升研究

摘要

深度学习网络的出现使得机器学习算法的预测准确性大幅提升，并且推动了机器学习算法在图像识别、自然语言处理等领域更广泛的应用，具有里程碑式的意义。然而深度学习网络对存储空间和计算资源的庞大需求，限制其在移动端、嵌入式等低功耗和存储有限的领域的应用。为了解决这个问题，众多深度网络压缩算法被提出。其中网络量化这种将网络权值参数和激活值由 32 位浮点数压缩成 1-2 个比特的方法，能够最大程度地实现存储空间上的压缩以及运算上的加速。但是网络量化会造成网络准确率的损失，如何设计合适的量化方法、网络结构以及训练算法，使得量化网络的准确率损失降到最低，是本文的主要研究内容。

本文首先从量化方法、网络训练以及比特运算三个方面构建了二值化和三值化网络，并且在 CIFAR-10 数据集上测试这两种量化网络的测试准确率和全精度网络的测试准确率之间的差距。其次，从增加少量的比特数且尽可能多地提升二值化网络测试准确率的角度出发，利用多比特能够提高网络性能和网络输出特征可分性逐层增强这两个特点，提出了量化比特由前往后逐层减少的混合比特量化网络结构。该网络结构的平均量化比特为 1 至 2 比特之间，然而在 CIFAR-10 和 CIFAR-100 这两个数据集上的测试准确率却能够远远超过二值化这种 1 比特网络，接近甚至好于三值化这种两比特网络。最后，研究了利用正则化和知识蒸馏将先验知识加入混合比特量化网络训练过程，以提升测试准确率的方案。实验结果显示，在 CIFAR-10 数据集上，知识蒸馏方法可以使学习能力较强的混合比特量化网络拥有更好的泛化性能，并且提升网络测试准确率 1%。

关键词：深度学习，网络量化，准确率提升，混合比特，知识蒸馏

Research On Modeling And Accuracy Improvement Of Hybrid-bit Quantized Neural Network

ABSTRACT

The emergence of deep learning dramatically improves the prediction accuracy and promotes the wide application in image recognition, natural language processing and other fields. However, its huge demand for storage space and computation resources, restricts its application in mobile terminals, embedded systems and other fields requiring low energy consumption and limited storage. In order to solve the problem, many deep learning compression algorithms have been proposed, among which network quantization, a method of compressing weight parameters and activations from 32-bits floats into 1 or 2 bits, achieves the maximum reduction of storage space and acceleration of computation. But Network quantization would result in the loss of network precision. How to design appropriate quantization method, network structure and training algorithm to minimize the loss of quantization is an important problem.

In this paper, from quantization method, network training and bit operation these three aspects, we firstly construct binarized and ternarized neural network. Using CIFAR-10 dataset, we obtain the difference between the testing precision of these two quantized network and that of full precision network. Secondly, from the point of view of improving the testing accuracy of binarized neural network through simply increasing a small amount of bits, we propose the hybrid-bit quantization network structure, whose quantization bits gradually decrease from front to back layer, taking advantage of the characteristic of network with several bits and the better classification ability for the output characteristic of latter layer. On the condition of using average quantization bits between 1 and 2, the network testing precision approaches or even surpasses that of network with two quantization bits in CIFAR-10 and CIFAR-100 dataset. Finally, we study how to add prior knowledge into the training of hybrid-bit quantized neural network with regularization and knowledge distillation, to improve the testing accuracy without adding additional bits. The experiment results show that knowledge distillation could make hybrid-bit quantized network who has better learning ability possess better generalized ability and improve the testing accuracy by nearly 1% in CIFAR-10 dataset.

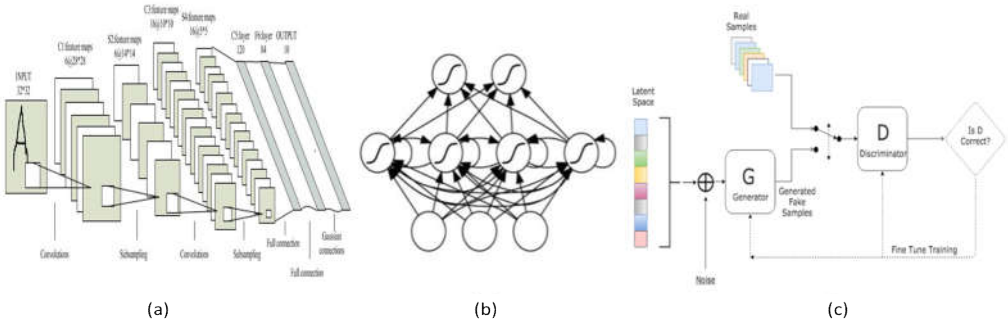
Key words: deep learning, network quantization, precision improvement, hybrid-bit, knowledge distillation

目 录

第一章 绪论

1.1 研究背景与意义

作为机器学习领域重要方法，深度学习凭借其强大的拟合能力，在许多方面优于现有其他机器学习算法，特别是在大型数据集上展现巨大优势。在 2015 年 ImageNet 挑战赛(ILSVRC2015)上， Kaiming He、Xiangyu Zhang 等人[1]提出的 152 层的残差深度网络结构(ResNet-152)以 80.62% 的测试集分类准确度，远超其他统计机器学习算法分类准确度 20%以上。卷积神经网络 (CNN) [2]、循环神经网络 (RNN) [3]以及对抗生成网络 (GAN) [4]这三种深度网络框架提出，使得最近几年深度学习在很多领域取得了广泛的应用，如：目标识别[5]、医学图像处理[6]、自然语言处理[7]和大数据挖掘[8]等。



(a) (b) (c)

图 1-1 CNN、RNN 以及 GAN 网络结构示意图
((a) CNN (b)RNN (c)GAN)

但是现阶段深度学习存在几个缺点，包括训练时对标记样本的需求量多、模型复杂使得训练耗费时间和精力以及容易陷入局部最优解等。其中一个严重制约深度学习广泛应用的因素是深度学习模型比较大，耗费计算机存储资源比较多，如：AlexNet 需要 200MB 内存，VGG 网络需要 500MB 内存以及 ResNet-101 需要 200MB 内存。此外深层卷积神经网络主要由大量乘加单元(MAC)组成，而普通处理器只能串行进行这种运算。如果让深度学习网络在普通处理器上进行运算，所用的时间难以想象。因此深度学习网络训练往往需要 GPU，但 GPU 功耗非常大，价格非常高昂，在移动端等功耗要求非常严苛领域不是最佳选择[9]。我们希望深度学习网络的训练和部署能够摆脱对于 GPU 设备依赖。然而，深度学习网络在功耗比较低，内存资源十分有限、运算能力也有限的设备，比如手机、可穿戴装置等移动端设备或者 CPU、FPGA 等嵌入式设备上，又难以直接开展部署。这就限制了深度学习在无人驾驶、机器人强化学习以及在线学习等领域的应用。

为了解决深度学习模型占用内存大和运算时间多的问题，许多深度学习模型压缩构想被提出，主要分为四类，包括参数共享和剪枝、低秩分解、卷积核映射以及知识蒸馏[10]。近几年来，为了进一步提升深度网络的时间空间性能，将深度网络中参数由 32 位浮点数通过合适的量化方式转化为 1、2 或者 8 个比特的方案应运而生，这就是后来所称的量化神经网络[11]。在存储空间上，量化神经网络所占用的内存空间最多为全浮点数网络的$\frac{1}{32}$；在运

算时间上，对于网络权值和输出量化将会把网络中的乘加运算转化为相应的位运算，大幅节约运算时间。量化网络的出现使得深度学习模型在 FPGA 等低功耗设备上部署甚至训练成为可能，同时也连接起当前最流行的深度学习算法和硬件之间桥梁，使研究者更多地从硬件角度思考如何对深度学习模型进行运算资源优化分配[9]。

1.2 国内外研究现状

1.2.1 深度网络压缩方法

在深度学习网络的量化方法出现之前，深度网络压缩方法主要包括网络剪枝、低秩分解、卷积核映射以及知识蒸馏。

网络剪枝是深度学习网络压缩领域最早被提出的方法，其核心思想是去除已经训练好的深度学习网络中冗余的权值参数，而这些权值参数的去除不会对于网络性能产生影响。这种方法在决策树、随机森林等机器学习算法中也有应用。早期的网络剪枝包括权值衰减[12],OBD(Optimal Brain Damage) [13]以及 OBS(Optimal Brain Surgeon) [14]。第一种方法单纯依据网络中权值大小来决定哪些权值连接需要被去除；后面两种方法计算损失函数对网络中权值参数的黑塞矩阵(Hessian Matrix)以及各个网络参数对于损失函数值的影响程度，影响程度较低的可以被去除掉。随后 Suraj Srinvas 等人[15]依据去除权值后对于网络输出影响程度最低的原则提出 data-free 的网络剪枝方法，其能够一次去除多个冗余的权值连接，去除效率比前面方法更高。Song Han 等人[16]将网络训练和网络剪枝结合起来，通过先剪枝后训练方法不断迭代获取最佳的网络权值连接方式。网络剪枝使得深度学习网络复杂度下降，在几乎不影响算法测试准确率的情况下实现深度学习网络的压缩。

卷积核是深度学习网络占用内存空间最大的部分，而低秩分解正是利用若干秩比较小的张量进行线性组合来逼近原有的张量方法，将秩大的卷积核分解成若干个秩小的卷积核，达到模型压缩加速的目的。Amos Sironi 等人[17]通过字典学习的方法学习分解后的小卷积核，并且提出秩较大的卷积核是秩较小的卷积核线性组合而成。Max Jaderberg 等人[18]结合不同矩阵分解的办法，使得深度网络运算速度提高 4.5 倍并且测试准确率只下降 1%。低秩分解思想也催生了两种新型轻量化网络结构诞生，分别是 MobileNet[19]和 SqueezeNet[20]。MobileNet 应用在对于时延要求特别小的移动设备上，主要特点是将标准卷积运算分解成深度卷积和逐点卷积，这样的操作使得网络运算量减少了 8 至 9 倍。而 SqueezeNet 则是将原有 3*3 卷积核部分替换成 1*1 卷积核，并且减少 3*3 卷积核的通道数，从而实现模型大幅度压缩。低秩分解能够实现模型的大幅度压缩和计算时间的大幅度减少，但是矩阵分解过程比较耗费时间，并且低秩分解目前还是逐层分解，不能做到对网络的统一分解。

核方法是机器学习领域常用方法，目的是将线性不可分的样本通过正定核映射转换到线性可分的空间中。类似核方法，卷积核映射利用卷积神经网络平移不变性以及权值共享等特点，将原有卷积核通过某种变换映射到新的空间，使得卷积核所需要的存储空间减小。Wenling Shang[21]以及 Hongyang Li[22]等人分别提出了利用负变换和多偏移量非线性变换的方式实现卷积核压缩。

知识蒸馏[23-25]是深度学习网络压缩领域比较新颖的方法，其核心思想是训练一个小的网络尽可能仿照一个大的网络。深度学习领域之父的 Hinton 率先提出这个概念，其将已经训练好的大网络的类 Softmax 输出作为待训练的小网络损失函数的一部分，这样使得大网络学习到的知识传递到小网络之中，从而提高小网络测试准确率。利用知识蒸馏方法，相近结构却性能不同的深度网络之间可以构成师生关系，学生网络通过学习老师网络输出的特点，尽

可能逼近老师网络。知识蒸馏方法也可以和其他深度学习网络压缩方法结合，从而提高其他深度学习网络压缩方法的性能。

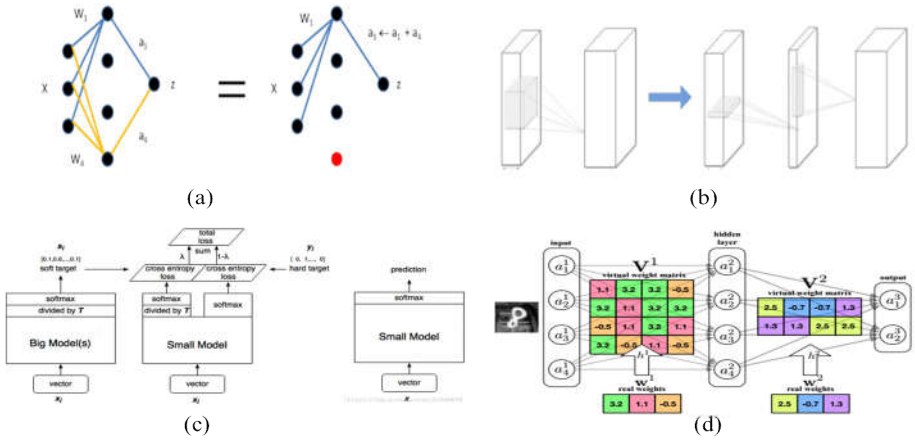


图 1-2 深度学习网络压缩方法

((a)网络剪枝方法 (b)低秩分解方法 (c)知识蒸馏方法 (d)网络量化哈希方法)

1.2.2 深度网络量化方法

深度学习量化方法与大多数深度学习网络压缩方法的思路不一样，大多数深度网络压缩方法都是去除网络的冗余项以及对于网络结构分解。而深度学习量化方法是一种权值共享的方法，往往深度学习量化方法能够在相同准确率损失下实现比其他深度网络压缩方法更高的压缩率。而且量化后的深度学习网络往往可以采用位运算这种方式，从而极大地提高了网络运算效率。相比于其他网络压缩方法，采用量化方式的网络运算时间往往最少。

表 1-1 不同网络压缩方法准确率、内存压缩以及运算速率提升比较[10]

网络压缩方法	ImageNet 上准确率	内存压缩	运算速率
网络剪枝	57.3% (AlexNet)	~9x	~9x
低秩分解	60.2%(MobileNet)	~7x	8-9x
卷积核映射	56.7%(AlexNet)	~4x	~2x
网络量化	44.2%(BNN AlexNet)	~32x	~58x

最早提出的深度学习量化方法是聚类量化。Song Han 等人[26]在网络训练过程中将权值矩阵中的值进行 K-Means 聚类，权值矩阵中的元素量化成几个聚类中心的值，然后对于聚类中心进行更新。Wenlin Chen[27]等人则利用哈希技巧和随机权值共享方法，实现量化。但聚类量化有两个方面缺点，首先是聚类运算复杂度很大，网络训练过程耗费时间很长；其次聚类量化后不能采用位操作，对于 FPGA 等硬件来说不是最佳选择。

许多研究者开始探究如何用公式来规定量化过程。Matthieu Courbariaux 等[28]提出二值化权值网络，在训练过程中将网络参数通过符号函数量化成+1 和-1，在前向计算和反向传播的过程中采用了二值化权值。网络权值参数由 32 位浮点数压缩成了 1 个比特，实现了网络规模 $\frac{1}{32}$ 的压缩。在二值化权值基础上，Itay Hubara 等[29]同时二值化激活值，这样将深度网络的乘加运算转化为位运算（XNOR-popcount 运算），极大提高运算速度。但采用二值化

后网络测试准确率会有一定程度衰减，为将准确率损失降到最低，Mohammad Rastegari 等[30]在每一层引入比例因子，并通过最优化方法推导比例因子表达式，称为 XNOR-Net。在二值化网络基础上，研究者提出了三值化网络[31,32]，三值化网络比二值化网络多了 0 这个量化选项，一方面完善了基本运算操作数，另一方面增加网络稀疏程度，在增加一个比特情况下提高网络性能。随后旷视科技的研究人员提出了多比特量化网络 DoReFa-Net[33]，并且对权值、激活值和导数分别设计量化方式。Itay Hubara 等人提出了不同于 DoReFa-Net 的量化方式[11]，测试集准确率比 DoReFa-Net 更高。

量化必然伴随准确率损失，怎么训练量化网络使得准确率损失降到最低也是研究重点。Wei Tang 等[34,35]在学习率、缩放因子以及正则化项等方面提出了如何训练二值化网络来获得更高准确率的建议。Ze ChunLiu 等提出类似 ResNet 结构的 Bi-Real 网络[36]，将二值化前的激活值通过 shortcut 连接与二值化后的激活值相加作为最终结果输出。Joseph Bethge 等[35]也验证了 shortcut 这种连接结构对量化网络的性能有提高。量化网络在参数更新阶段采用全精度权值参数，但在下一代训练的前向传播中权值参数又会被量化。基于这个特点，清华大学的 Guoqi Li 团队[37]提出在权值更新过程中不使用全精度权值，而是采用状态转移的方法在不同量化数值之间进行转换，在训练过程中不需要储存全精度权值，节省内存。

1.3 论文研究内容与结构

毕业设计课题的研究集中在量化网络的算法层面，包括网络的量化方式、量化网络的训练过程以及量化网络与其他方法的结合。研究问题是在保证网络压缩 16~32 分之一的情况下，通过设计量化方式、调整网络结构、改进训练过程的方法，来提升量化网络准确率。毕业设计课题不涉及具体量化后的网络如何在 FPGA 等硬件设备上进行调度部署。毕业设计工作如下：

(1) 根据前人提出的二值化和三值化网络的量化方法和训练方法，构建二值化和三值化网络模型，并且在 CIFAR-10 数据集上训练二值化和三值化网络，使其测试集准确率超过 85%。

(2) 复现多比特量化网络量化组合的实验结果，研究了不同层特征输出可分性的变化情况。依据这两个实验结果提出了混合比特量化网络结构。并在 CIFAR-10 和 CIFAR-100 上训练混合比特量化网络，并将其准确率、模型体积和运算时间与二值化和三值化网络的进行对比。

(3) 设计二值化和三值化网络正则化项，并且在 CIFAR-10 数据集上训练带正则化项和不带正则化项的量化网络，利用权值参数分布直方图和准确率曲线分析正则化对量化网络性能的影响。

(4) 研究了如何将知识蒸馏方法和量化网络结合，尝试利用知识蒸馏提升量化网络准确率，在 CIFAR-10 数据集上从准确率和类别两个角度，分析了知识蒸馏参数、量化网络学习能力对知识蒸馏方法提升量化网络性能的程度的影响。

论文的章节安排如下：

第二章主要介绍量化网络中最基本的二值化和三值化网络结构。从量化方式、网络训练方法以及比特运算三个角度构建二值化和三值化网络。并在 CIFAR-10 数据集上从准确率、模型体积以及运算时间三个角度进行全精度、二值化和三值化网络对比实验，研究比例系数和稀疏度对准确率影响。最后介绍了二值化和三值化网络正则化项设计过程，并进行了系列对比实验。

第三章从实验理论基础、多比特量化方式、网络训练和比特运算四个角度，诠释我们提出的混合比特量化网络的由来、训练和部署。最后通过与二值化和三值化网络在准确率、网络大小以及运算时间对比实验，测试混合比特网络在准确率、网络大小以及运算时间三个方面性能。

第四章阐释知识蒸馏方法如何与量化网络结合，以提升量化网络准确率。通过准确率实验和类别实验，测试知识蒸馏方法对量化网络性能提升作用以及分析其中影响因素。

第二章　二值化和三值化网络

二值化和三值化网络是量化网络最简单的形式，即分别将深度学习网络中的参数量化成一个或者两个比特。这种量化程度很大的深度网络也有一定的理论依据，首先深度学习的优化算法中有随机梯度下降（SGD），即每一次迭代选取一部分梯度来更新权值参数。虽然每一代并不是走最优方向，但每一次总和会逐渐趋向于全局最优点。量化网络在每一次迭代过程会给网络带来一定量化误差，而这种量化误差具有一定随机性，即量化网络在每一代中不一定走梯度最优方向，但经过若干迭代过程后也能够趋近于较优值。其次深度网络不需要在训练数据上拟合得很好，为了防止过拟合，可以在网络中会添加 dropout 层[2]，在训练过程中随机丢弃一些连接，增加网络稀疏度。量化网络，尤其是含有 0 这个量化值的量化网络，某种意义上来说就带有一定正则化功能。这种功能使得量化网络相比于全精度网络而言，有更好的泛化能力。

2.1 二值化神经网络

2.1.1 量化方式

二值化量化网络的量化方式分为两种，第一种是确定型量化方式，即根据权值参数特征将其具体量化至某个值。第二种是随机型量化方式，即利用权值参数求取量化到某个值的概率，根据概率随机分配量化值。

符号函数是二值化确定型量化方式最常用的方法，如公式(2-1)所示，根据深度网络权值与激活值的正负号决定量化数值，正数量化至+1，负数量化至-1。

$$\mathrm{binarize}(x)=\begin{cases}-1 & x<0\\ +1 & x\geq 0\end{cases} \tag{2-1}$$

直接利用参数正负号进行量化会带来比较大的误差，如果在每一层加上比例系数，使得量化前后权值矩阵和输出激活值向量尽可能接近，量化误差将会减小。

定义二值化前一个卷积核张量为$\mathbf{W}=\{W_{ij}\}_{m\times n\times h}$，卷积核对应的量化前该层输入部分为$\mathbf{X}=\{X_{ij}\}_{m\times n\times h}$。把卷积核和对应输入矩阵展成向量，定义为$\mathbf{w},\mathbf{x}\in\mathbf{R}^{m\times n\times h}$。则卷积核矩阵以及对应输入部分的乘加运算等价于展成向量后的内积。定义量化后的权值向量和输入向量为$\mathbf{b},\mathbf{h}\in\{+1,-1\}^{m\times n\times h}$，权值向量和输入向量的比例系数为$\beta$和$\alpha$。量化前后权值向量和输入向量对应元素相乘所得到的向量尽可能接近，即转化为下面优化问题：

$$\alpha^{*},\beta^{*},\mathbf{b}^{*},\mathbf{h}^{*}=\underset{\alpha,\beta,\mathbf{b},\mathbf{h}}{\arg\min}\ \|\mathbf{x}\odot\mathbf{w}-\beta\alpha\mathbf{h}\odot\mathbf{b}\|_2^2 \tag{2-2}$$

其中⊙运算指的是两个向量之间对应元素相乘。定义$\mathbf{y} \in \mathbf{R}^{m\times n\times h}$ 且$y_i = x_i w_i$ ，$\mathbf{c} \in \{-1,+1\}^{m\times n\times h}$ 且$c_i = h_i b_i$ ，$\gamma = \beta\alpha$ ，则(2-2)的优化问题转化为：

$$\begin{aligned} J(\mathbf{c},\gamma) &= \| \mathbf{y} - \gamma \mathbf{c} \|_2^2 \\ \gamma^*, \mathbf{c}^* &= \arg\min_{\gamma,\mathbf{c}} J(\mathbf{c},\gamma) \end{aligned} \tag{2-3}$$

展开(2-3)，我们可以得到：

$$J(\mathbf{c},\gamma) = \mathbf{y}^T\mathbf{y} - 2\gamma\mathbf{y}^T\mathbf{c} + \gamma^2\mathbf{c}^T\mathbf{c} \tag{2-4}$$

由于$\mathbf{c}\in\{-1,+1\}^{m\times n\times h}$，则$\mathbf{c}^T\mathbf{c} = m\times n\times h$ ；而且 $\mathbf{y}$ 是已知的向量，则$\mathbf{y}^T\mathbf{y}$ 是常数。令$const = \mathbf{y}^T\mathbf{y}$ ，则(2-4)公式可以写成：

$$J(\mathbf{c},\gamma) = const - 2\gamma\mathbf{y}^T\mathbf{c} + \gamma^2 m n h \tag{2-5}$$

我们可以通过解(2-6)优化问题得到 $\mathbf{c}$ 的最优解：

$$\mathbf{c}^* = \arg\max_{\mathbf{c}} \{\mathbf{y}^T\mathbf{c}\} \quad s.t.\ \mathbf{c} \in \{+1,-1\}^{m\times n\times h} \tag{2-6}$$

当 y_i 是正数时，c_i 取+1。当 y_i 是负数时，c_i 取-1。此时(2-6)优化问题取到最大值。则 $\mathbf{c}$ 的最优解为：

$$\mathbf{c}^* = sign(\mathbf{y}) = sign(\mathbf{x}) \odot sign(\mathbf{w}) \tag{2-7}$$

(2-5)优化函数对于 γ 求导，可以得到推导的 γ 最优化

$$\gamma^* = \frac{\sum|y_i|}{m\times n\times h} = \frac{\sum|x_i w_i|}{m\times n\times h} \approx \left(\frac{1}{m\times n\times h}\|\mathbf{x}\|_{l1}\right)\left(\frac{1}{m\times n\times h}\|\mathbf{w}\|_{l1}\right) \tag{2-8}$$

因此，对于含有比例系数的二值化网络，对于权值矩阵和输出激活值矩阵量化方法依然是符号函数，而比例系数是权值矩阵和对应卷积输入矩阵各个元素绝对值的均值。这个不仅对于深度网络卷积层适用，对线性层也适用。

上述两种方法是确定型二值量化方法，对于每个权值参数和输出激活值来说量化值是确定的。公式(2-9)展示的是随机二值量化方式：

$$\begin{aligned} binarize(x) &= \begin{cases} +1 & with\ probability\ p = \sigma(x) \\ -1 & with\ probability\ p = 1-\sigma(x) \end{cases} \\ \sigma(x) &= clip\left(\frac{x+1}{2},0,1\right) = \max\left(0,\min\left(1,\frac{x+1}{2}\right)\right) \end{aligned} \tag{2-9}$$

随机二值量化方式先计算量化为+1 概率 σ(x)，如果 x 是正数，量化为+1 的概率较高；相反 x 是负数，量化为-1 概率较高。与前面所提到的量化方式不同，在(-1,+1)区间中的每一个取值都有一定机率被量化成+1 或者-1。由于随机二量化方式求导比较困难，因此后续所提到量化方式均采用确定型量化方式。

2.2.2 网络训练过程

量化和网络训练结合方式也有两种类型。第一种是对于已经训练好的模型进行量化，这样使得在全精度网络情况下预测值和真实标签之间的距离最小，但这种方法往往会带来比较大的量化误差，而设计合适最小化量化误差算法也比较困难。第二种是在训练过程中加入量化过程，在前向传播和反向求导过程中均使用量化参数，这样一方面最小化预测值和真实标签之间的距离，另一方面也可以最小化量化误差。

由于网络权值参数以及网路激活值使用的是+1 和-1 的离散值，反向求导过程中涉及到对于离散值求导。量化后的激活值是量化前激活值取符号函数，而符号函数在大部分定义域中的取值处的导数为 0，这会带来严重的梯度消失情况。为了解决这个问题，需要重新模拟符号函数的导数。符号函数导数理论表达式为：

$$\frac{d(binarize(x))}{dx}=\frac{d(sign(x))}{dx}=\frac{d(2u(x)-1)}{dx}=2\delta(x) \tag{2-10}$$

(2-10)中 u(x)为单位阶跃函数，符号函数导数从理论上为 2*δ(x)的冲激函数，对于冲激函数的模拟是符号函数导数重新表述的关键。冲激函数有两个很重要性质：

$$\begin{cases}\int_{-\infty}^{+\infty}\delta(x)dx=1\\ \delta(x)=0 \quad x\neq 0\end{cases} \tag{2-11}$$

我们希望二值量化函数的导数尽量能够逼近冲激函数，满足冲激函数一些性质。因此矩形脉冲模拟和三角形脉冲模拟方案被提出。

$$\frac{d(binarize(x))}{dx}\approx u(x-1)-u(x+1) \tag{2-12}$$

$$\frac{d(binarize(x))}{dx}\approx(2-2|x|)(u(x+1)-u(x-1)) \tag{2-13}$$

图 2-1 展示了二值量化函数、二值量化函数理论导数以及二值量化函数导数模拟函数图像。

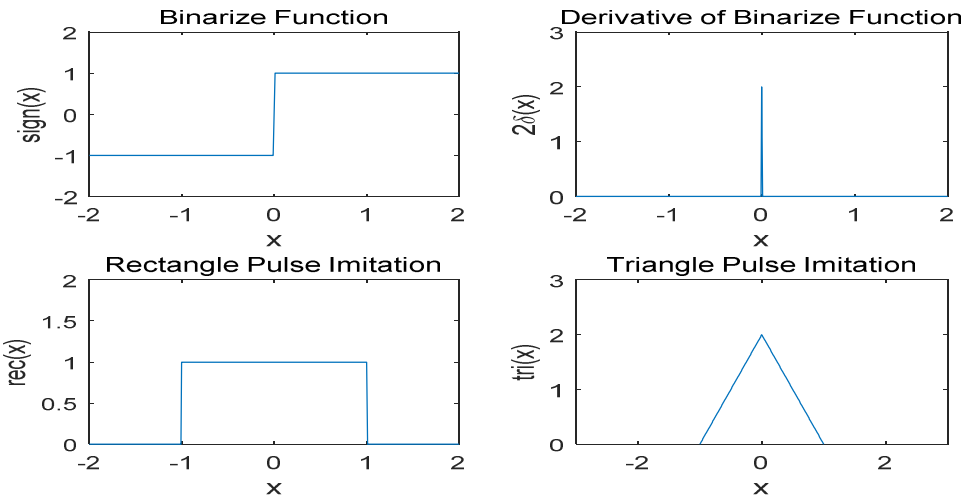


图 2-1 二值化量化函数、量化函数导数以及模拟函数图

因此定义 g_q 为损失函数对于某一层二值化后输出激活值导数，g_r 为损失函数对于相同层量化前输出激活值导数，根据二值量化函数导数模拟函数推导以及链式法则，有：

$$g_r = g_q \frac{dbinarize(r)}{dr} = g_q 1_{|r|\le 1} \tag{2-14}$$

因此在二值化网络某一层实数激活值量化成二值化激活值过程之前，需要限制在区间[-1,1]里面，使得反向求导过程能够按照(2-14)方式进行。在二值量化网络中，采用激活函数 Hardtanh 可以解决这个问题。

$$Hard\tanh(x) = Clip(x,-1,1) = \begin{cases} -1 & x \le -1 \\ x & -1 < x < 1 \\ 1 & x \ge 1 \end{cases} \tag{2-15}$$

网络中实数权值参数也被限制在范围内，因为绝对值比 1 大很多的权值参数最终在前向传播过程中也会被量化成-1 和+1 两个值，对于实数权值参数范围限制对量化网络性能没有影响。

二值化深度网络前向传播、反向求导和权值更新伪代码如下：

Algorithm 1 二值化神经网络算法

Require: 一个batch的输入数据和对应标签(a_0,a*)，上次迭代后的网络权值参数W，上次迭代过程归一化层参数θ，上次迭代过程学习率η 和学习率衰减参数λ

Ensure: 当前迭代更新后权值参数W^{t+1}，更新后归一化层参数θ^{t+1}，以及更新后的学习率η^{t+1}

1.前向传播

for $k = 1\ to\ L$ **do**
  $W_k^b \leftarrow Binarize(W_k)$
  $s_k \leftarrow a_{k-1}^b W_k^b$
  $a_k \leftarrow BatchNorm(s_k, \theta_k)$
  if $k < L$ **then**
    $a_k^b \leftarrow Binarize(a_k)$
  end if
end for

2.反向求导

已知最后一层输出a_L以及数据标签a*，计算损失函数对于最后一层输出值的梯度$g_{a_L} = \frac{\partial Loss}{\partial a_L}$。

for $k = L\ to\ 1$ **do**
  if $k < L$ **then**
    $g_{a_k} \leftarrow g_{a_k}{}^b 1_{|a_k|}$
  end if
  $(g_{s_k}, g_{\theta_k}) \leftarrow BackBatchNorm(g_{a_k}, s_k, \theta_k)$
  $g_{a_{k-1}^b} \leftarrow g_{s_k} W_k{}^b$
  $g_{W_k^b} \leftarrow g_{s_k}{}^T a_{k-1}^b$
end for

3.参数更新

for $k = 1\ to\ L$ **do**
  $\theta_k^{t+1} \leftarrow Update(\theta_k, \eta, g_{W_k^b})$
  $W_k^{t+1} \leftarrow Clip(Update(W_k, \eta, g_{W_k^b}), -1, 1)$
  $\eta^{t+1} \leftarrow \lambda\eta$
end for

可以发现，二值化网络训练前向传播和反向求导过程采用的是二值化权值参数和输出激活值，而在二值化网络参数更新过程采用的是实数权值参数。其实实数权值参数也不是一定要求的。Guoqi Li 团队[37]提出用状态转移方法从而避免实数权值使用。在模型最终部署中网络权值参数为量化后的权值参数。

带有比例系数二值化网络与不带比例系数二值化网络在训练过程中大体上一致。区别主要在于前向传播过程。在前向传播过程中，对于卷积层而言，根据公式(2-8)先计算输入比例系数矩阵和卷积核比例系数，然后对于卷积核以及卷积层输入取符号函数，二者进行卷积操作后再与输入比例系数矩阵进行对应元素相乘运算，最后乘以卷积核比例系数。这样获得该二值化网络一个卷积核运算下的特征映射。对于线性层而言，先计算的是权值比例系数向量和输入比例系数，后面运算与卷积层类似。卷积层运算过程可见图 2-2。

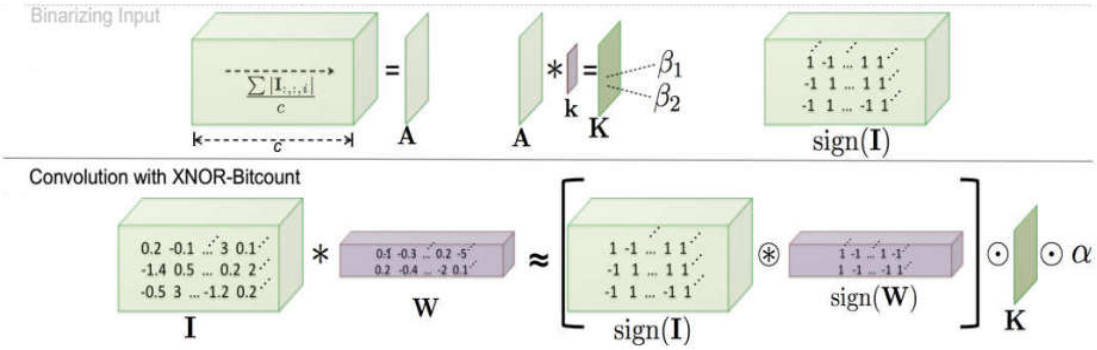


图 2-2 带比例系数的二值化网络卷积层运算过程[30]

2.1.3 网络比特运算

量化成+1 和-1 的二值化网络究竟如何在支持 0 和 1 的比特运算的硬件设备上进行部署，关键在于二值化网络最终如何利用 0 和 1 进行运算。

由于训练完网络权值参数取值为-1 和+1，硬件设备主要进行 0 和 1 比特运算。中间还需要运算转化过程。一般情况下，我们使取值-1 对应硬件运算比特 0，取值+1 对应硬件运算比特 1。定义运算转化后权值向量为 $\tilde{\mathbf{w}}$ ，激活值向量为 $\tilde{\mathbf{x}}$ ；运算转化前权值向量为 $\mathbf{w}$ ，激活值向量为 $\mathbf{x}$ 。其中 $\tilde{\mathbf{w}}$, $\tilde{\mathbf{x}} \in \{0,1\}^n$ ， $\mathbf{w}$, $\mathbf{x} \in \{-1,1\}^n$ 。

表 2-1 列出的是转化前乘积运算和转化后同或运算对应情况。

表 2-1 二值化量化前乘积运算和量化后同或运算对应关系

x_i	w_i	$x_i \bullet w_i$
-1	-1	1
-1	1	-1
1	-1	-1
1	1	1

$\tilde{x}_i$	$\tilde{w}_i$	$\tilde{x}_i \odot \tilde{w}_i$
0	0	1
0	1	0
1	0	0
1	1	1

表格中关系可以用(2-16)式子表示：

$$x_i \bullet w_i = \begin{cases} \tilde{x}_i \odot \tilde{w}_i & \tilde{x}_i \odot \tilde{w}_i = 1 \\ \tilde{x}_i \odot \tilde{w}_i - 1 & \tilde{x}_i \odot \tilde{w}_i = 0 \end{cases} \tag{2-16}$$

假设$\tilde{\mathbf{x}} \odot \tilde{\mathbf{w}}$结果中前 m 维元素为 0，其余为 1。则公式(2-17)展示如何利用二进制权值和输入数据计算取值为-1 和+1 的权值和输入数据乘加运算：

$$\begin{aligned} \mathbf{w}^T\mathbf{x} &= \sum_{i=1}^{n} w_i x_i = \sum_{i=1}^{m} (\tilde{w}_i \odot \tilde{x}_i - 1) + \sum_{i=m+1}^{n} (\tilde{w}_i \odot \tilde{x}_i) \\ &= -m + popcount(\tilde{\mathbf{w}} \odot \tilde{\mathbf{x}}) = -(n - popcount(\tilde{\mathbf{w}} \odot \tilde{\mathbf{x}})) + popcount(\tilde{\mathbf{w}} \odot \tilde{\mathbf{x}}) \\ &= 2 * popcount(\tilde{\mathbf{w}} \odot \tilde{\mathbf{x}}) - n \end{aligned} \tag{2-17}$$

对于带有比例系数的二值化网络，(2-17)转变成如下公式：

$$\mathbf{w}^T\mathbf{x} = 2 * \alpha * \beta * popcount(\tilde{\mathbf{w}} \odot \tilde{\mathbf{x}}) - n * \alpha * \beta \tag{2-18}$$

popcount($\tilde{\mathbf{x}} \odot \tilde{\mathbf{w}}$)指的是计算$\tilde{\mathbf{x}} \odot \tilde{\mathbf{w}}$ 中 1 的个数，很多研究者针对 popcount 设计了高效低耗的算法[9,38,39]，部分硬件设备上也有 popcnt 指令。二值化量化方法使得前向传输中的乘加运算转化为同或和 popcount 的位运算，运算效率提升 58 倍，并且更好地在硬件设备上对深度学习网络进行资源部署。

2.2 三值化神经网络

相比于二值化神经网络，三值化神经网络多了 0 这个量化值，其他量化值与二值化神经网络相同。因此三值化神经网络也被称作稀疏二值化神经网络。在硬件部署上，三值化神经网络比二值化神经网络多了一个比特，是两比特网络，在存储和运算量上略有增加。但凭借运算能力增强以及适当稀疏性这两个特点，三值化网络测试准确率显著好于二值化网络。

2.2.1 量化方式

三值化神经网络在量化过程中引入了阈值，在正负阈值范围内网络参数和激活值被量化成 0，大于正阈值网络的参数和激活值被量化成+1，小于负阈值的网络参数和激活值被量化成-1。公式(2-19)展示三值化量化方式。

$$ternarize(x) = \begin{cases} 1 & x \geq \Delta \\ 0 & -\Delta < x < \Delta \\ -1 & x \leq -\Delta \end{cases} \tag{2-19}$$

其中 Δ 代表三值化网络阈值，其也控制了网络稀疏区间取值范围。对于网络每一层 Δ 的计算如下：

$$\Delta = t \times \max(|\mathbf{x}|) \tag{2-20}$$

x 代表网络权值或者激活值向量，t 是比例因子。由于 **x** 实数值最终会被限制在[-1,1]中，网络稀疏度主要由比例因子 t 控制，可以近似理解网络稀疏度就为比例因子 t。

同样三值化神经网络也可以引入比例系数来增强网络性能，与二值化量化网络不相同的是其对于大于正阈值和小于负阈值这两个区间分别引入比例系数。如公式(2-21)：

$$ternarize(x) = \begin{cases} C^p & x \geq \Delta \\ 0 & -\Delta < x < \Delta \\ -C^n & x \leq -\Delta \end{cases} \tag{2-21}$$

式子中 C^P 和 C^N 为非负实数。在三值化网络中，比例系数是通过神经网络反向求导和参数更新过程自动学习获得的，和二值化网络通过解最优化方法推导求得的方式不一样。

2.2.2 网络训练过程

反向求导过程与二值化网络基本类似。由于三值量化函数是双阈值函数，其求导后为在两个阈值处的单位脉冲函数。类比二值化网络进行矩形脉冲近似，三值量化函数的导数可以近似成以两个阈值为中心，面积为 1 的矩形脉冲。由于对于量化函数的导数近似是否特别准确对于量化网络最后性能影响不大[35]，三值量化函数的导数可以近似认为和二值量化函数导数相同，即把每一层激活值输出限制在[-1,1]中，反向求导值为 1。

$$\frac{d(ternarize(x))}{dx} = 1_{|x|\leq 1} \tag{2-22}$$

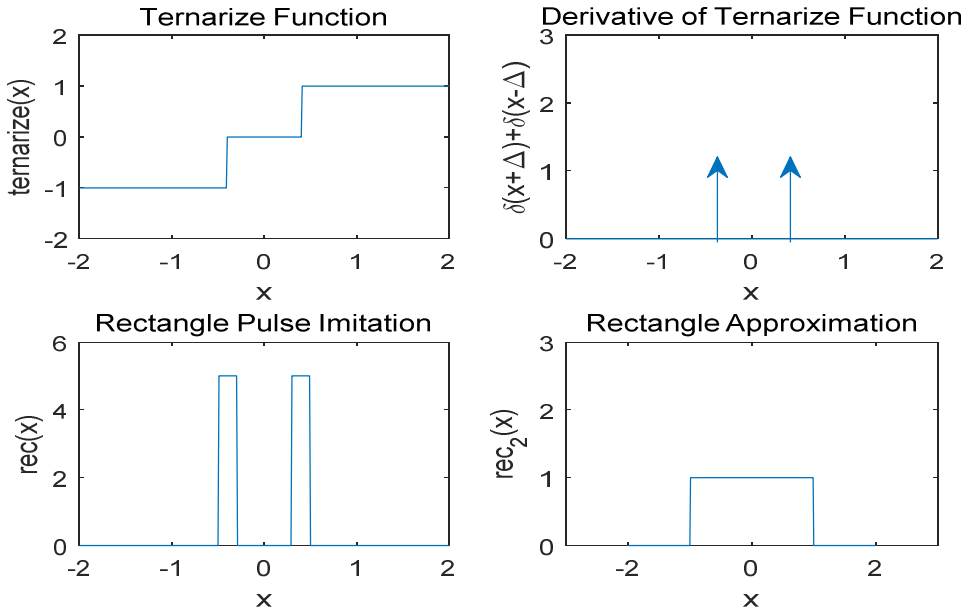


图 2-3 三值化量化函数、量化函数导数以及矩形脉冲模拟图
（(3）双矩形脉冲模拟 (4) 单矩形脉冲逼近-用于训练）

对于带有比例系数三值化网络，由于每个网络层都有比例系数，在反向求导和参数更新

过程中，每层比例系数都需要更新。损失函数对权值或者激活值的正负比例系数的求导公式如下：

$$\frac{\partial L}{\partial C^{p}}=\sum_{i\in I_l^{p}}\frac{\partial L}{\partial x^{t}(i)},\frac{\partial L}{\partial C^{n}}=\sum_{i\in I_l^{n}}\frac{\partial L}{\partial x^{t}(i)} \tag{2-23}$$

其中 x 可以是网络激活值或者权值，带有上标 t 的变量表示三值化后的网络激活值或者权值，$I_l^p=\{i\,|\,x_l>\Delta_l\},I_l^n=\{i\,|\,x_l<-\Delta_l\}$ 。损失函数对于实数权值或者激活值求导公式如下：

$$\frac{\partial L}{\partial x_l}=\begin{cases}W_l^{p}\dfrac{\partial L}{\partial x_l^{t}} & x_l>\Delta_l\\ \dfrac{\partial L}{\partial x_l^{t}} & -\Delta_l\le x_l\le\Delta_l\\ W_l^{n}\dfrac{\partial L}{\partial x_l^{t}} & x_l<-\Delta_l\end{cases} \tag{2-24}$$

带有比例系数的三值化网络在训练过程中和二值化网络主要区别是根据公式(2-21)加入比例系数进行三值量化，在反向传播过程中根据(2-23)以及(2-24)求取网络损失函数对网络权值和激活值比例系数和本身实数值的导数。

2.2.3 网络比特运算

三值化后神经网络在硬件上需要两比特表示，其中一个比特表示符号，另外一个比特表示数值。下面研究如何利用硬件的两比特运算来还原三值化神经网络前向传播过程。

设硬件上表示的两比特权值为 $\mathbf{W_1W_0}$，表示的两比特激活值为 $\mathbf{A_1A_0}$。其中 $\mathbf{W_1}$、$\mathbf{W_2}$、$\mathbf{A_1}$、$\mathbf{A_2}$ 为$\{0,1\}^n$。三值化后神经网络权值为 **W**，激活值为 **A**。它们为$\{-1,0,1\}^n$。设在硬件上运算结果为 $\mathbf{R_1R_0}$。规定两比特数最高位为符号位，低比特位为数值位。如果为正数，高比特位为 1；如果是负数，高比特位为 0；如果结果为 0，高比特位既可以取 1，也可以取 0。则表 2-2 列出的是两种运算之间转换关系：

表 2-2 三值化运算转换前后关系

W_i	A_i	$W_i * A_i$
-1	0	0
1	0	0
0	0	0
-1	-1	1
1	-1	-1
0	-1	0
-1	1	-1
1	1	1
0	1	0

$W_{1i}W_{0i}$	$A_{1i}A_{0i}$	R_{1i}	R_{0i}	$R_{1i}R_{0i}$
0 1	0(1) 0	0(1)	0	0,2
1 1	0(1) 0	0(1)	0	0,2
0(1) 0	0(1) 0	0(1)	0	0,2
0 1	0 1	1	1	3
1 1	0 1	0	1	1
0(1) 0	0 1	0(1)	0	0,2
0 1	1 1	0	1	1
1 1	1 1	1	1	3
0(1) 0	1 1	0(1)	0	0,2

利用卡诺图，可以推导二进制输出每一位与二进制权值和激活值的关系。

A1A0 / W1W0	00	01	11	10
00				
01		1	0	
11		0	1	
10				

A1A0 / W1W0	00	01	11	10
00				
01		1	1	
11		1	1	
10				

图 2-4 运算转换后三值化输出与三值化权值和激活值输入关系
（左侧为 R_{1i}，右侧为 R_{0i}，图中圆圈为卡诺图化简过程）

则如果对于结果为 0 的数，高比特位只考虑为 0。我们可以推得运算转换后三值化输出与权值和激活值输入关系公式如下:

$$R_{0i} = W_{0i}A_{0i} \tag{2-25}$$
$$R_{1i} = (\overline{W}_{1i}\overline{A}_{1i} + W_{1i}A_{1i})R_{0i} = (W_{1i} \odot A_{1i})W_{0i}A_{0i}$$

运算转换前权值向量和激活值向量做内积结果中含有 1 和-1 元素个数为：

$$count(1) = popcount((\mathbf{W}_0 \odot \mathbf{A}_0)\mathbf{W}_1\mathbf{A}_1) \tag{2-26}$$
$$count(-1) = popcount(\mathbf{W}_1\mathbf{A}_1) - popcount((\mathbf{W}_0 \odot \mathbf{A}_0)\mathbf{W}_1\mathbf{A}_1)$$

运算转换前权值向量和激活值向量内积结果和运算转换后位运算之间关系为：

$$\mathbf{W}^T\mathbf{A} = count(1) - count(-1) \tag{2-27}$$
$$= 2*popcount((\mathbf{W}_0 \odot \mathbf{A}_0)\mathbf{W}_1\mathbf{A}_1) - popcount(\mathbf{W}_1\mathbf{A}_1)$$

运算转换前三值化网络乘加运算最终转化为同或、与和 popcount 运算，类似二值化网络分析，网络运算速度大大提高，并且便于在硬件上进行资源调度。三值化网络运算比二值化网络运算稍微复杂，但是准确率上却带来一定提升。

2.3 二值和三值网络正则化设计

相对于全精度网络，量化网络学习能力弱，过拟合情况没有全精度网络严重。但是在 VGG 和 Resnet 网络训练的过程中，训练准确率和测试准确率在经过一段迭代过程后还是会出现一定距离的差距，出现一定程度的过拟合情况。本节对二值化网络和三值化网络的正则化设计进行研究，探索如何针对量化网络设计合适的正则化项。

2.3.1 二值化网络正则化项

对于深度学习网络，常见正则化项为 L1 和 L2 正则化项。L1 正则化项是权值向量的 1-范数，L2 正则化项是权值向量的 2-范数，其公式如下：

$$L1(\mathbf{w}) = ||\mathbf{w}||_1^1 = \sum_{i=1}^{N} |w_i|, L2(\mathbf{w}) = ||\mathbf{w}||_2^2 = \sqrt{\sum_{i=1}^{N} w_i^2} \tag{2-28}$$

L1 范数和 L2 范数目的是使网络参数尽可能往 0 方向靠拢，形成标准正态分布，使得网络稀疏化。但是对于量化网络而言，因为网络量化值是有限的，添加 L1 和 L2 范数正则化项会使得网络过于稀疏，无法充分利用网络量化值，网络性能会下降。因此，有必要对于量化网络设计新的正则化项。

其中朴素想法是让量化网络的量化值作为正则化项的根，极小化正则化项就是使得网络参数靠近几个量化值。对于二值化网络，这种多项式正则化项为：

$$Poly(\mathbf{w}) = \sum_{i=1}^{N} (1 - w_i^2) \tag{2-29}$$

其中 w_i 的取值范围为[-1,1]。对于实数权值来说，只有当取+1 和-1 的时候量化惩罚为 0，取其他值时候量化惩罚为正数。这样使得量化前实数权值参数尽可能靠近-1 和+1 两个量化值。

由于在前向传播和反向求导过程中我们使用的是量化权值，如果直接将由实数权值求取的正则化项加在损失函数中，会带来问题。因此正则化因素加在反向求导的过程中，对于二值化多项式正则化项导数如下：

$$\frac{\partial(Poly(\mathbf{w}))}{\partial w_i} = -2w_i \tag{2-30}$$

则二值化多项式损失函数对于某个权值参数导数为：

$$\begin{aligned}\frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}},\mathbf{w}))}{\partial w_i} &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} + \frac{\partial(Poly(\mathbf{w}))}{\partial w_i} \\ &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} - 2w_i\end{aligned} \tag{2-31}$$

损失函数对于量化网络中实数权值导数分为两项，第一项是网络输出和样本标签之间的损失函数对于网络实数权值参数的导数，第二项是正则化项对于网络实数权值导数。二值化网络多项式正则化项的式子及其导函数的图像如下：

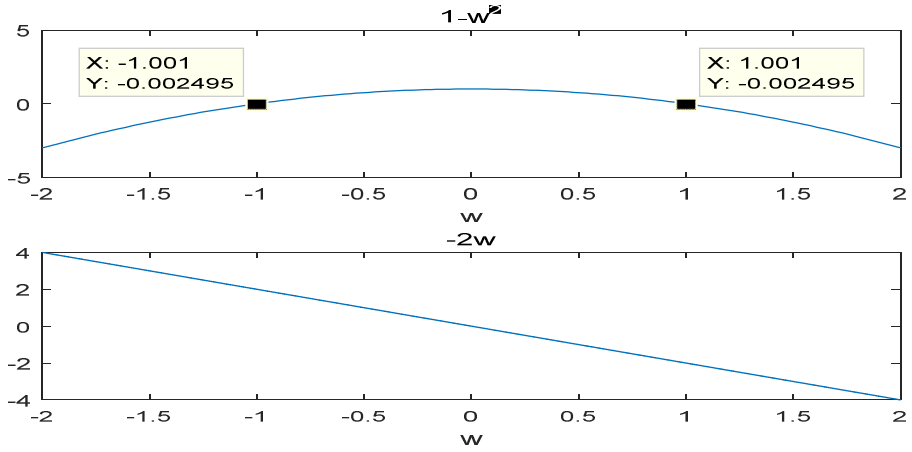


图 2-5 二值化网络多项式正则化项及其导数图像

2.3.2 三值化网络正则化项

三值化网络正则化项的多项式形式与二值化网络类似，目标是使得网络趋近于 0、-1 和 +1 三个值。一方面使得网络尽可能稀疏，另一方面使得网络量化误差不会太大。

$$Poly(\mathbf{w}) = \sum_{i=1}^{N} A w_i^2 (1 - w_i^2) \tag{2-32}$$

添加进入损失函数后对实数权值求导为：

$$\begin{aligned}\frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}},\mathbf{w}))}{\partial w_i} &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} + \lambda \frac{\partial(Poly(\mathbf{w}))}{\partial w_i} \\ &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} + \lambda A(2w_i - 4w_i^3)\end{aligned} \tag{2-33}$$

三值化网络正则化项还有周期函数形式，其原理和多项式正则化项类似。

$$Period(\mathbf{w}) = \sum_{i=1}^{N} A \sin^2(\Pi w_i) \tag{2-34}$$

添加进入损失函数后对实数权值求导为：

$$\begin{aligned}\frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}},\mathbf{w}))}{\partial w_i} &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} + \lambda \frac{\partial(Poly(\mathbf{w}))}{\partial w_i} \\ &= \frac{\partial(Loss(\mathbf{z},\hat{\mathbf{z}}))}{\partial w_i} + 2A\lambda\Pi\sin(\Pi w_i)\cos(\Pi w_i)\end{aligned} \tag{2-35}$$

图 2-5 展示了三值化网络的正则化项的多项式和周期函数这两种形式的原函数和导数图像。我们可以看到这两种正则化方式的缺陷是其对于实数权值的导数有不是在-1,1 和 0 这三个量化点处的零点。这意味着当权值参数取非量化点处的值但在该点处正则化项的导数为 0 的时候，正则化项不能使这些权值参数往-1、+1 和 0 这三个量化点方向靠拢。三值化正则化的预想是当取值离量化点很远的时候应该要给比较大的惩罚，对应导数绝对值应该比较大；相反对应导数绝对值应该比较小。因此文章中提出了第三种三值化正则化方法。

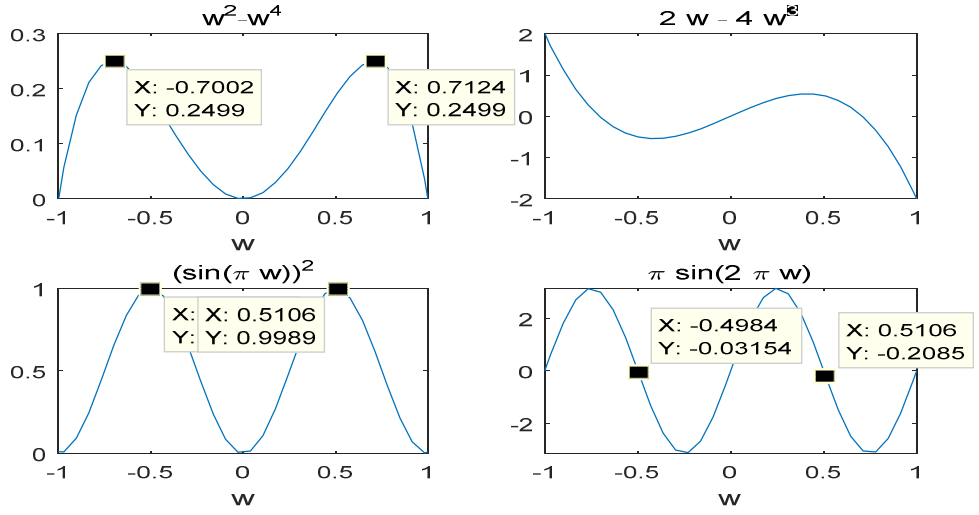


图 2-6 三值化网络多项式正则化、周期函数正则化原函数与导数图像

第三种正则化方案是从正则化项导函数角度设计的。我们以阈值为 0.5 的三值化网络为例，根据我们的预想，当权值取值为±0.5 时，正则化导数绝对值最大。而当权值为-1、0 和 1 时，正则化导数绝对值为 0。则该正则化项的导数为：

$$my\_grad(w_i)=\begin{cases} A\sin(\Pi w_i) & -0.5\le w_i\le 0.5 \\ -A\sin(\Pi w_i) & -1\le w_i<-0.5 \ or\ 0.5<w_i\le 1 \end{cases} \tag{2-36}$$

对于正则化项导数积分并且对于网络所有正则化项求和，则该正则化项的表达式为：

$$my\_regularizer(w_i)=\begin{cases} \frac{A}{\Pi}\cos(\Pi w_i)+\frac{A}{\Pi} & -1\le w_i<-0.5 \ or\ 0.5<w_i\le 1 \\ -\frac{A}{\Pi}\cos(\Pi w_i)+\frac{A}{\Pi} & -0.5\le w_i\le 0.5 \end{cases} \tag{2-37}$$

$$my\_regularizer(\mathbf{w})=\sum_{i=1}^{N} my\_regularize(w_i)$$

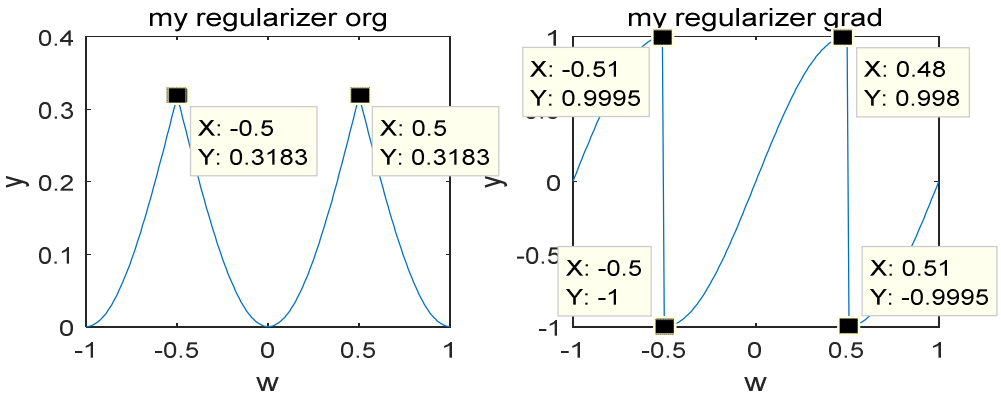


图 2-7 第三种三值化网络正则化项原函数与导数图像

2.4 实验及结果

2.4.1 实验环境和数据集

本文使用的数据集有两种，均为公开标准数据集。第一种是 CIFAR-10 数据集，其由 60000 张 RGB 彩色图片构成，每张图片为 32*32 像素。数据集总共分为 10 类，包括飞机、鸟、卡车等。其被分为 5 个训练集合和 1 个测试集合，每一个数据集合包含 10000 张图片[45]。训练集合是从 10 个类别中随机 1000 张图片组成的；剩余 10000 张图片构成测试集。第二种是 CIFAR-100 数据集，其和 CIFAR-10 共享一个数据集，分为 100 个类，100 个类别中分为 20 个大类，每个大类下有 5 个小类别。训练集和测试集的划分和 CIFAR10 一致。

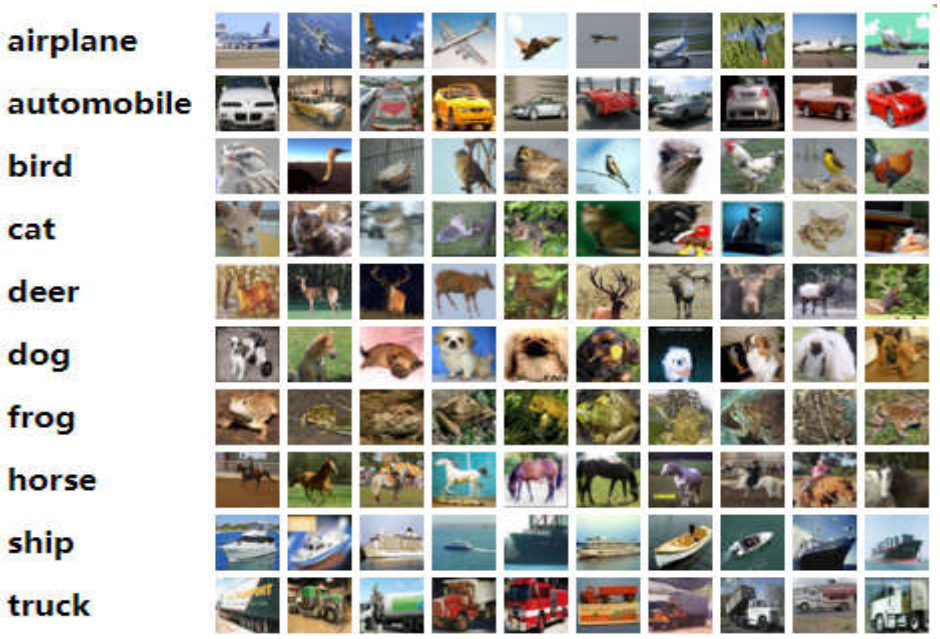
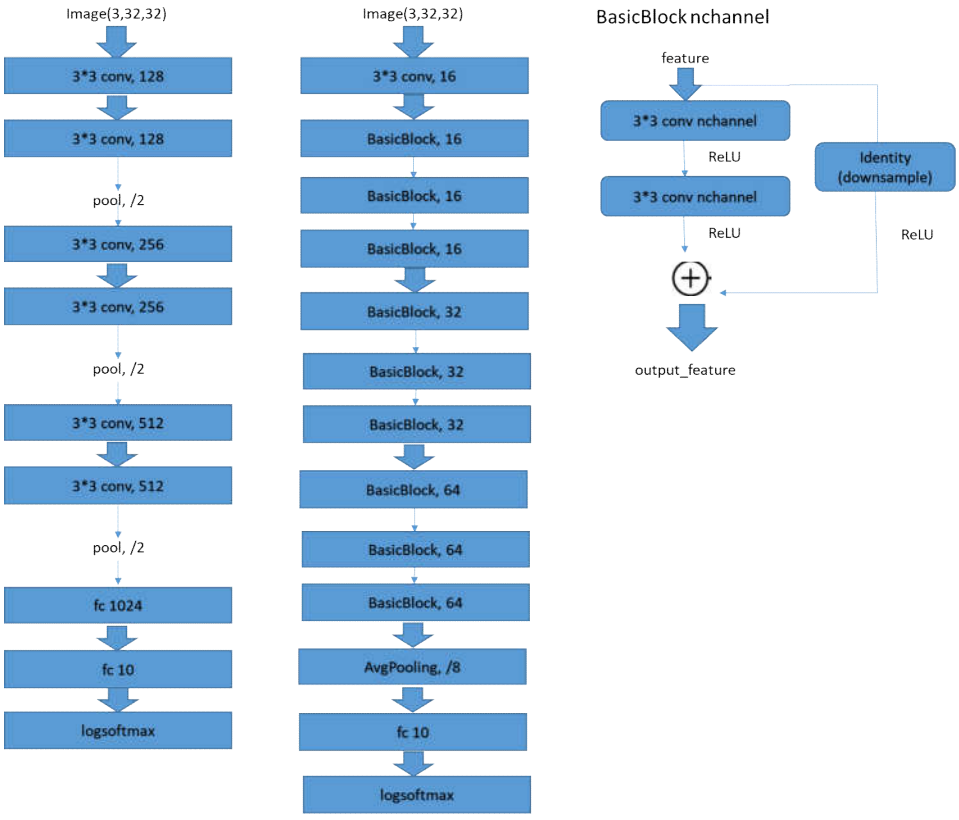


图 2-8 CIFAR-10 每个类别部分样本

量化网络部署的最终目标是要在移动端或者嵌入式设备上，但在初期训练和算法研究阶段还是利用 GPU 设备。实验中所使用的是 NVIDIA 公司的 GeForce GTX 1080 Ti 类型的 GPU。实验操作系统是 Ubuntu16.04，编程语言为 Python3.6，实验中采用了深度学习框架为 Pytorch，版本为 0.4.1。其与其他深度学习框架最大区别是支持动态图构建和运算，更方便地让深度学习研究者进行算法开发。

2.4.2 实验使用的网络结构

为了与全精度网络以及目前量化网络领域其他的量化方法及网络训练方法对标，本文所有实验使用的均为 VGG-like 和 ResNet 这两种经典网络结构。VGG 网络[40]由牛津大学的 Visual Geometry Group 提出，其在 ILSVRC2014 的图像分类竞赛中获得第二名。在实验中使用得 VGG 网络和最初提出的 VGG 网络结构上有部分区别，因此我们称之为 VGG-like 网络。而 ResNet 网络[1]由 Microsoft 团队的 Kaiming He 等人提出，其在 ILSVRC2015 的图像分类竞赛中获得冠军，并且分类准确率远远高于其他网络结构。在 CIFAR-10 和 CIFAR-100 实验中，我们使用的是 ResNet-20 这种网络结构。实验中使用的全精度的 VGG-like 和 ResNet 网络结构如下图所示：



(a) (b)

图 2-9 实验中用到 VGG-like 和 ResNet 网络结构示意图
((a) VGG-like 网络 (b)ResNet-20 网络及其中 BasicBlock 模块)

对于经典网络结构，量化网络和全精度网络主要区别在卷积层、线性层运算以及激活函数使用。下图展示的是全精度网络和量化网络结构上区别体现。

图 2-10 全精度网络和量化网络在网络结构上差异

量化网络某层特征在输入卷积层前先被量化，后和量化的卷积核做卷积。得到结果经过池化层和归一化层后，最后通过 Hardtanh 函数激活。在全精度网路中，没有量化过程，并且最后激活函数为 ReLU。值得注意一点是，网络第一层的输入不用被量化，第一层输入被量化会带来比较大的准确率损失。

2.4.3 二值化、三值化与全精度网络对比实验

这部分主要比较了二值化和三值化两种网络的性能特点，以及二值化和三值化网络中是否加上比例系数对最后网络性能的影响。

首先在 CIFAR-10 数据集上训练全精度、二值化和三值化的 VGG-like 网络，比较三种情况下 VGG-like 网络能够达到的最高测试准确率。三种网络训练过程中 batch 的大小为 100，训练迭代次数为 200 次，训练时采用的优化算法是 Adam，在整个训练过程中没有添加正则化项，训练的学习率使用 warm up+exponential decay 的变学习率的方式，前 20 个迭代周期学习率线性增加，到达最大值；后面 180 个迭代周期学习率指数衰减。对于全精度网络学习率而言学习率最大值为 0.02，对于二值化网络而言学习率最大值为 0.03，对于三值化网络而言学习率最大值为 0.02。后 180 个迭代周期学习率的指数衰减公式为：

$$lr(n) = lr_{\text{in}\,it} * gamma^{n-warm\_up} \quad n \in [warm\_up+1, epochs] \qquad (2\text{-}38)$$
$$gamma = 0.001^{1/(epochs-warm\_up)}$$

这种学习率设置的优点是在训练开始的时候加大学习率可以跳过部分局部最小点，并且加快迭代训练过程；而在训练中后期逐渐减小学习率有利于趋近于全局最小值，减小在全局最小值附近的震荡。

下面是 VGG-like 的全精度网络、二值化网络以及三值化网络在 CIFAR-10 上测试集上所能够达到的最高准确率：

表 2-3 全精度、二值化和三值化 VGG-like 网络在 CIFAR-10 上测试准确率

float	binary	ternary
0.9353	0.8987	0.9156

下面的图为二值化网络的训练和测试损失函数曲线以及全精度、二值化、三值化 VGG-like 网络的测试准确率曲线。

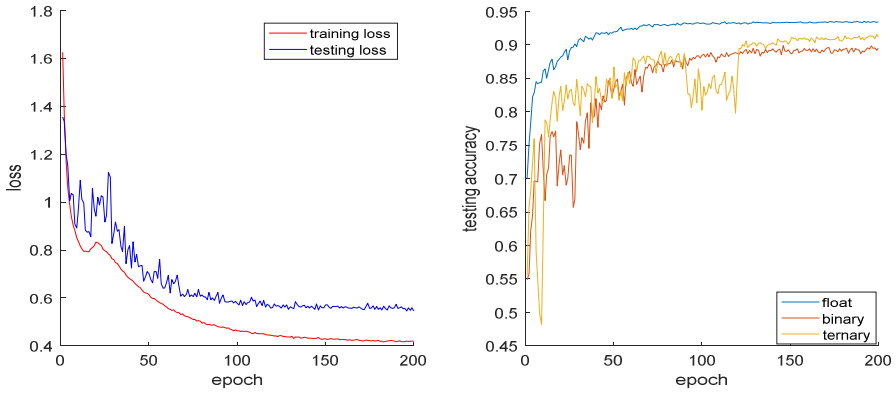


图 2-11 全精度、二值化与三值化 VGG-like 网络损失函数和测试准确率图
（左侧：二值化网络损失函数曲线 右侧：全精度、二值化与三值化测试准确率曲线）

由 2-10 的曲线可以看出，相对于全精度网络，量化网络会带来一定准确率上的损失，但在 CIFAR-10 这种小规模数据集上，准确率损失不大。相对于二值量化网络，三值量化网络在网络准确率上更优，这是由于三值量化网络中的量化比特增加使得网络表达能力增强，而稀疏区间加入使得网络的泛化能力更强。下面挑选 CIFAR-10 测试集中的 6 个样本，观察其在全精度网络和量化网络中各层的特征输出情况。

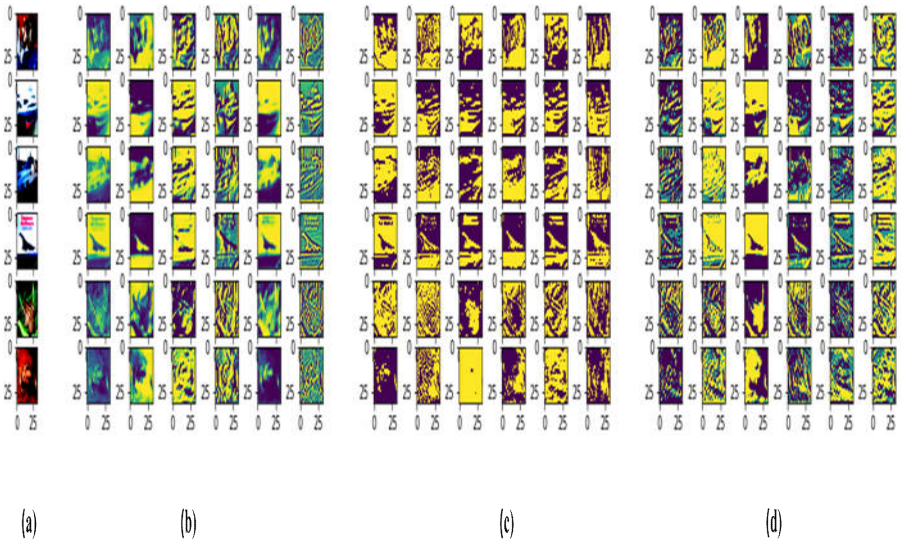


图 2-12 CIFAR-10 数据上全精度、二值化以及三值化 VGG-like 网络第一层特征输出图
((a)RGB 图片 (b)全精度网络 (c)二值化网络 (d)三值化网络)

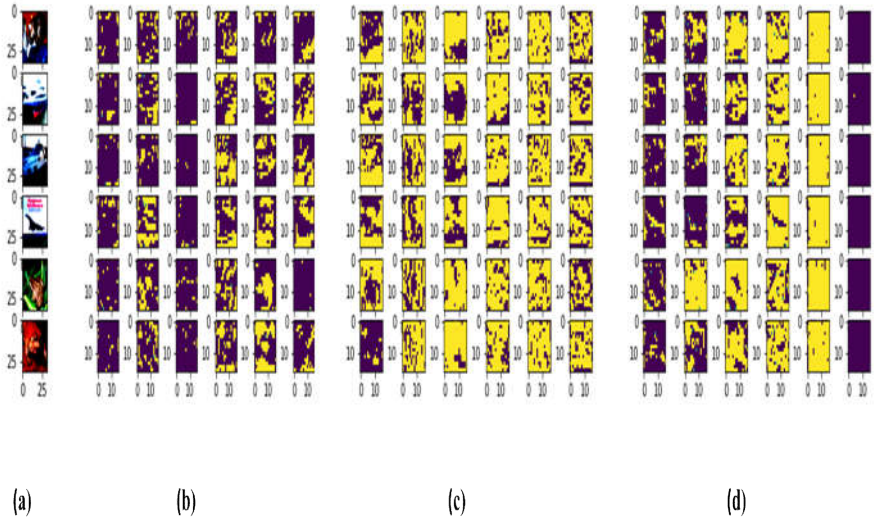


图 2-13 CIFAR-10 数据上全精度、二值化以及三值化 VGG-like 网络第七层特征输出图
((a)RGB 图片 (b)全精度网络 (c)二值化网络 (d)三值化网络)

图 2-11 和 2-12 选取了汽车、飞机、青蛙等六张图片输入全精度、二值化和三值化 VGG-like 网络，然后输出中间层的特征。中间层的输出先通过 Sigmoid 函数，把输出范围限制在[0,1]之间，再乘以 255，输出范围限制在[0,255]。根据分析，全精度网络可以取到[0,255]中整数值，而二值化网络只取得到其中两个灰度值，三值化取得到其中三个灰度值。可以看到二值化和三值化网络能够提取图像中边沿轮廓。这个和在数字图像课程中学习的二值分割有异曲同工之妙。但是全精度网络在第一层提取的特征更细致，并且其能够提取图片中的色彩变化信息，这些会导致后面层的特征可分性增强。这也是全精度网络测试准确率比二值化

及三值化网络高的原因。三值化网络比二值化网络更好，也是因为前层特征提取更细致，使得后层特征可分性增强。

网络大小和运算时间是量化网络相对于全精度网络的优势所在，利用这部分提到的 VGG-like 网络结构，对比二值化、三值化以及全精度网络在网络大小和运算时间上的性能。

表 2-4 CIFAR-10 数据集上全精度、二值化以及三值化 VGG-like 网络体积大小

	After Compression (MB)	*Actual Compression Rate*	*Ideal Compression Rate*
Float	51.90	x1.0	x1
Binary	1.70	x30.5	x32
Ternary	3.55	x14.6	x16

表 2-5 CIFAR-10 数据集上全精度、二值化以及三值化 VGG-like 网络测试耗费时间

	Time Consumption (s)	*Actual Acceleration Rate*	*Ideal Acceleration Rate*
Float	7.300	x1	x1
Binary	0.146	x50	x58
Ternary	0.243	x30	x36

可以看到，二值化和三值化网络在模型存储以及运算速度上有比较显著优势，限制二值化和三值化网络的核心依然是准确率损失。因此我们需要思考如何在几乎不牺牲存储和运算速度的基础上提升二值化和三值化网络识别准确率。

2.4.4 比例系数和稀疏度参数影响实验

比例系数对于量化网络性能影响是其中的一个研究内容，下面比较了 CIFAR-10 数据集下二值化网络和三值化 VGG-like 网络不加比例系数与添加比例系数测试准确率。其中其他参数控制一致。二值化网络最大学习率统一为 0.03，三值化网络最大学习率统一为 0.02。

表 2-6 二值化和三值化 VGG-like 网络（添加或不加比例系数）测试准确率

Binary(with coefficient)	*0.8990*
Binary(without coefficient)	0.8987
Ternary(with coefficient)	0.9073
Ternary(without coefficient)	0.9156

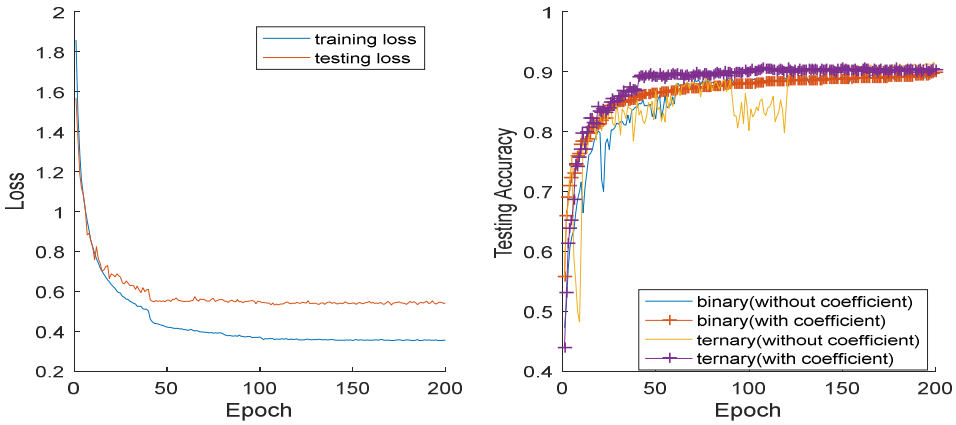


图 2-14 二值化和三值化带系数和不带系数 VGG-like 网络损失函数和测试准确率图（左侧：带系数三值化网络损失函数图 右侧：二值化和三值化带系数和不带系数 VGG-like 网络测试准确率图）

添加比例系数对于量化网络性能的影响有限，有时添加比例系数的网络测试准确率甚至不如没有比例系数的网络。我们分析了在量化网络中比例系数影响被减弱原因，主要是网络中的归一化层[41]使得比例系数影响没有那么重要。

设不加比例系数某层网络输出为 X，添加比例系数后网络输出为 αX，则若这一层的下一层为归一化层，则添加比例系数后网络输出的均值和方差为：

$$\begin{aligned} E(\alpha X) &= \alpha E(X) \\ D(\alpha X) &= \alpha^2 D(X) \end{aligned} \tag{2-39}$$

经过归一化层后，输出变成：

$$\frac{\alpha X - E(\alpha X)}{\sqrt{D(\alpha X)}} = \frac{\alpha X - \alpha \mathrm{E}(\mathrm{X})}{\sqrt{\alpha^2 D(X)}} = \frac{X - E(X)}{\sqrt{D(X)}} \tag{2-40}$$

带有比例系数的量化网络和不带比例系数的量化网络在归一化层后输出是一致的，这也证明对于每层为单比例系数网络，比例系数是否添加对于网络性能提高没有影响。同样也可以证明对于每层为多比例系数网络，由于归一化层存在，比例系数对于网络性能提升不显著。

稀疏度是带有稀疏区间的量化网络（包括三值化网络）重要属性，而该量化网络稀疏性主要由网络稀疏区间宽度（阈值）所决定的，由于稀疏阈值计算是利用一个比例系数乘以实数权值矩阵的最大值，本节中稀疏度和比例系数基本上等价的。本节研究主要通过变化三值化网络中稀疏区间的阈值，探索三值化网络性能和网络稀疏度关系。

下图为三值化网络在 CIFAR-10 数据集上的测试准确率和网络稀疏度之间关系，对于不同网络稀疏度，控制其他网络参数不变(batch size=100, optimizer:Adam, largest_lr=0.02, weight_decay=0)。

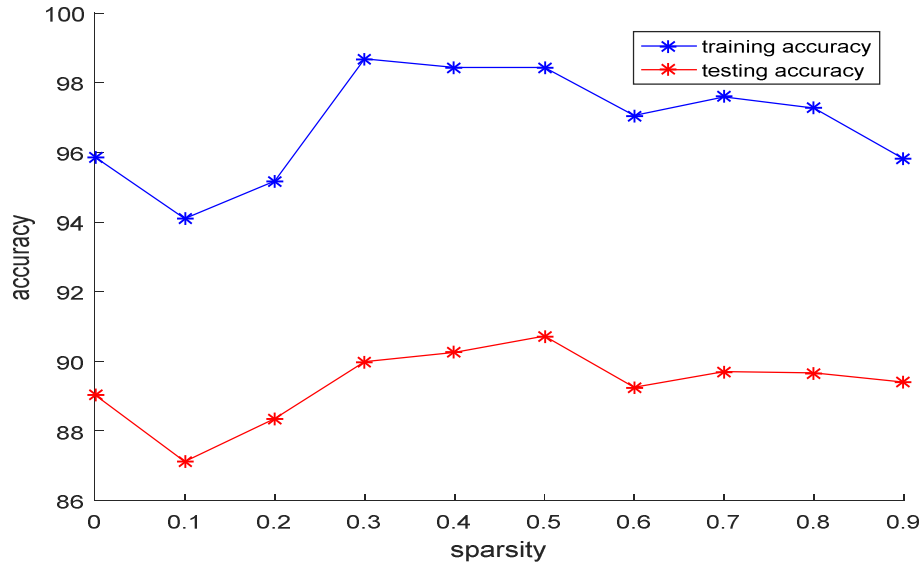


图 2-15 CIFAR-10 数据集上三值化 VGG-like 网络测试准确率与网络稀疏度关系曲线

从图 2-13，可以发现随着网络稀疏度的增加，三值化 VGG-like 网络测试准确率先增加后降低。开始的时候网络测试准确率增加原因是网络稀疏度增加使得网络泛化性能增强，在训练数据集上的过拟合情况减弱，在测试数据集上准确率提升。这个和在网络中添加 dropout 的原理类似。但随着网络稀疏度继续增加，网络逐渐变得过于稀疏，这使得网络学习性能下降，因而网络测试准确率下降。网络训练准确率在稀疏度为 0.3 的时候开始下降，而网络测试准确率在稀疏度为 0.5 时候开始下降。这也证明网络拟合能力并不和泛化能力相对应，而泛化能力真正反应网络真实的性能。网络最优稀疏度为 0.5。

2.4.5 二值化和三值化网络正则化系列实验

二值化和三值化网络正则化项的理论设计在 2.3 节中已经加以阐述，本节通过利用 CIFAR-10 数据集，在训练二值化和三值化的 VGG-like 网络过程中，添加不同种类的正则化项（L2 正则化、多项式正则化、周期函数正则化、周期函数导数正则化），研究在不同正则化情况下训练准确率和测试准确率之间差距以及测试准确率情况，探索量化网络中的正则化效应。

首先研究二值化网络中的正则化因素，实验中研究了不加正则化项的二值化网络、加 L2-正则化项的二值化网络以及加多项式正则化项的二值化网络在 CIFAR-10 上的测试准确率。实验中的正则化系数为 10^{-7}，其他参数均保持一致。(batch size=100, optimizer:Adam, largest_lr=0.03, weight_decay=0) 。下表为实验结果。

表 2-7 不同正则化项对于二值化网络测试准确率影响（正则化系数为 10^{-7}）

Regularizer	Training Precision	Testing Precision	Gap
None	0.9495	0.8987	0.0508
L2	0.9546	0.8890	0.0656
Polynomial	0.9306	0.8849	0.0457

可以看到，二值网络在添加多项式正则化项后，训练准确率和测试准确率之间差距有一定程度减小，但多项式正则化项不能够使网络测试准确率提升。通过分析，其实正则化项对于二值化网络几乎没有作用，这主要是缘于二值化网络没有稀疏区间。原本不靠近+1 或-1 的量化值也会被量化为+1 或-1。正则化项并不能使得二值化网络更加稀疏，不能使其泛化能力更强。

三值化网络中含有稀疏区间，正则化项理应对于三值化网络性能有一定提高，下面实验着重研究 L2 正则化、多项式正则化项、周期函数正则化项以及周期函数导数正则化项对于网络性能影响。实验中正则化系数为 10^{-7}，控制所施加正则化因素最大值相同，即多项式正则化比例系数为 4，正弦正则化项比例系数为 1，正弦导数正则化项比例系数为 $\pi/2$。其他参数均保持一致。(batch size=100, optimizer:Adam, largest_lr=0.02, weight_decay=0)，则实验结果如下所示：

表 2-8 不同正则化项类型对于三值化网络测试准确率影响（正则化系数为 $5*10^{-7}$）

Regularizer	Training Precision	Testing Precision	Gap
None	0.9814	0.9077	0.0737
L2	0.9481	0.8947	0.0534
Polynomial	0.9288	0.8914	0.0374
Periodic	0.9265	0.8887	0.0378
Periodic Grad	0.9446	0.8946	0.0500

对于同一种正则化项类型，施加不同正则化系数，调整原有损失函数和正则化项之间的权重，比较正则化系数对于三值化网络测试准确率影响。下面实验对于三值化 VGG-like 网络施加相同正则化项（周期函数导数正则化），改变正则化系数。其他实验条件和上面实验一致。

表 2-9 不同正则化系数对于三值化网络测试准确率影响（正则化项为周期函数导数）

Regularizer	Training Precision	Testing Precision	Gap
5e-5	0.9068	0.8814	0.0254
5e-6	0.9105	0.8850	0.0255
5e-7	0.9446	0.8946	0.0500
1e-7	0.9733	0.9066	0.0667
5e-8	0.9786	0.9053	0.0733

作出添加正则化进行训练后三值化网络实数权值参数的分布情况，如下图所示：

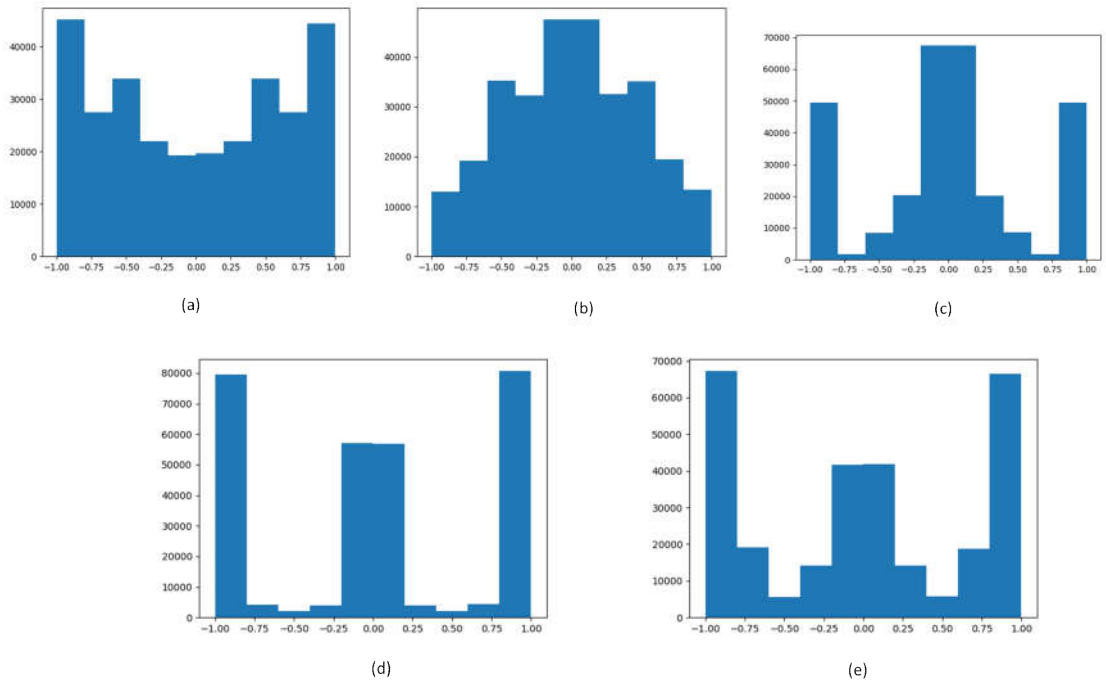


图 2-16 三值化网络加入不同正则化项类型后 VGG-like 网络第 7 层权值参数分布图
((a) 不加正则化 (b) L2 正则化 (c) 多项式正则化
(d) 周期函数正则化 (e) 周期函数导数正则化)

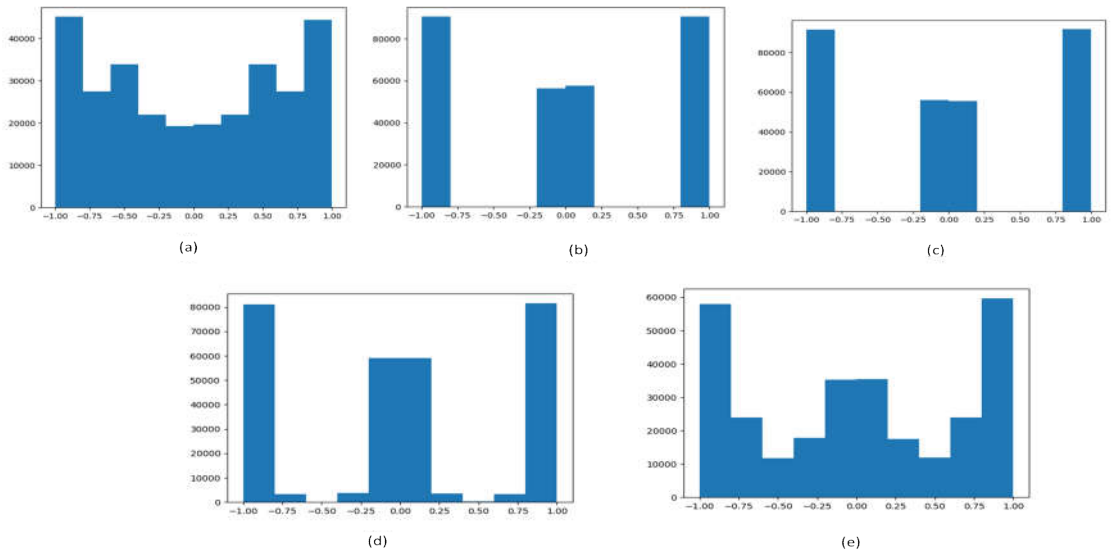


图 2-17 周期函数导数正则化后加入不同正则化系数后 VGG-like 网络权值参数分布图
((a) 不加正则化 (b) 5e-5 (c) 5e-6 (d) 5e-7 (e) 5e-8)

可以看到，加入 L2 正则化项使得网络权值更多权值参数分布在 0 附近，而加入多项式正则化、周期函数正则化以及周期函数导数正则化项可以使权值参数分布在 0、1 和-1 附近。加入正则化项后，参数分布能够达到我们的预想效果。加入不同正则化项后网络实值参数分

布有所不同，周期函数正则化项正则化效果比较强，几乎在三个量化离散值之间没有参数分布，而其他正则化项添加后，量化值之间存在一定参数分布。随着正则化系数减小，在离散值之间参数所占比例越来越大。然而，正则化对于测试准确率提升却不明显，增大正则化效果，只是使得训练准确率下降，测试准确率也随之下降。这个现象的一种解释是我们对量化网络性能达到最优时权值参数分布假设并不一定正确，因此我们需要寻找更好先验知识的办法。

2.5 本章小结

本章主要从量化方式、训练方法和前向比特运算三个方面，构建了二值化和三值化网络。在量化方式上，主要比较带比例系数和不带比例系数这两种量化方式，通过实验分析说明比例系数对于量化网络最后性能影响不大。在训练方法上，主要分析反向求导过程，反向求导中带有阶跃的函数的导数的模拟是关键。在前向比特运算方面，推导了{-1,1}({-1,0,1})集合与{0,1}集合之间运算对应关系，诠释乘加运算如何转化为位运算。最后利用量化前参数值尽可能靠近几个量化值的设想，提出了二值化和三值化的正则化项形式。

实验部分主要介绍论文中用到数据集、实验环境和实验用到网络结构，进行全精度网络、二值化网络和三值化网络准确率、模型体积以及运算时间三个方面的性能对比。研究比例系数和稀疏度参数对于网络性能影响，证明一定稀疏度可以提高量化网络性能。最后研究添加正则化项对二值化和三值化网络影响，发现设计的正则化项能够使网络权值分布符合我们预想，但不能够提高网络测试准确率，说明我们设计正则化依据的先验知识不一定正确，我们需要研究更好寻找正确先验知识办法。这部分是下部分提出的混合比特量化网络的基础和原因，混合比特量化网络的二值化和三值化部分与本章的一致。

第三章 混合比特量化网络

上一章主要阐述了量化网络中的二值化和三值化网络，深度网络中的参数被量化成为一个或者两个比特。在这种方案中，对于网络中所有的层，量化比特数目均保持一致。而实际中深度网络每一层对于量化比特要求不一定一样。利用这个特点，我们可以给深度网络每一层分配不同比特数，在允许一定准确率损失情况下，使得网络压缩率尽可能大。在某些关键网络层，量化比特数多；在某些不关键的网络层，量化比特数少。这就是第三章提出的混合比特量化网络模型。通过混合比特量化网络模型，在很小准确率损失下，网络平均比特数可以量化至 1 比特至 2 比特之间。

3.1 理论基础

这一节主要讲述混合比特量化网络的想法为什么被提出，通过两个实验为混合比特量化网络模型提供背景前提和理论支撑。第一个实验是对于不同网络权值、激活值以及导数的量化组合下网络特性的研究，阐明使用更多的比特能够使网络的准确率得到提升；第二个实验是对于网络每一层输出特征可分性的研究，通过这个实验提出网络每层比特具体怎么分布的构想。

3.1.1 多比特量化网络量化组合研究

对于网络的量化包括对于网络权值、网络激活值和网络导数量化，本小节主要研究不同量化比特组合带来影响。其中采用的量化方法是旷视公司所提出的的 DoReFa-Net 中量化方式[33]。我们在 CIFAR-10 和 CIFAR-100 两个数据集上测试了在不同量化组合情况下 Alex 网络特性[2]，并作出了 30 个迭代周期下网络测试准确率随着训练迭代次数变化曲线。实验中其他因素均保持不变，学习率变化采用的是在前 5 个迭代周期增加，最大值为 10^{-3}。在第 10 个迭代周期和第 20 个迭代周期分别减小为 $5*10^{-4}$ 和 10^{-4}。

其中图例中表示数字组合的第一位是权值量化比特，第二位是激活值量化比特，第三位是导数量化比特。可以发现，当导数量化比特小于 4 时，网络准确率衰减很大；当导数量化比特大于等于 6 比特时，导数量化位数增加给网络性能带来影响有限，此时影响网络性能的主要是网络权值和激活值的量化比特。网络权值和激活值量化比特增加，特别在数据量更大、类别更多的数据集上，带给网络性能提升会更显著。在小规模数据集上，量化位数对于网络准确率影响不大，任意比特网络基本上都能收敛到接近的值；而在大规模数据集上，不同量化位数的网络之间差距逐步拉大。因此网络比特数目对于网络性能是关键因素。

但采用多比特方式网络量化方法，随着量化比特数目增多，网络体积增大。我们想在尽可能很小地增加网络体积前提下利用多比特网络优点，因此想出混合比特网络结构。混合比特网络比特数最大值为多少是一个关键问题，根据 Itay Hubara 的实验结果[11]，8 比特网络的准确率接近全精度网络准确率，因此网络比特数最大值选取为 8 比特。

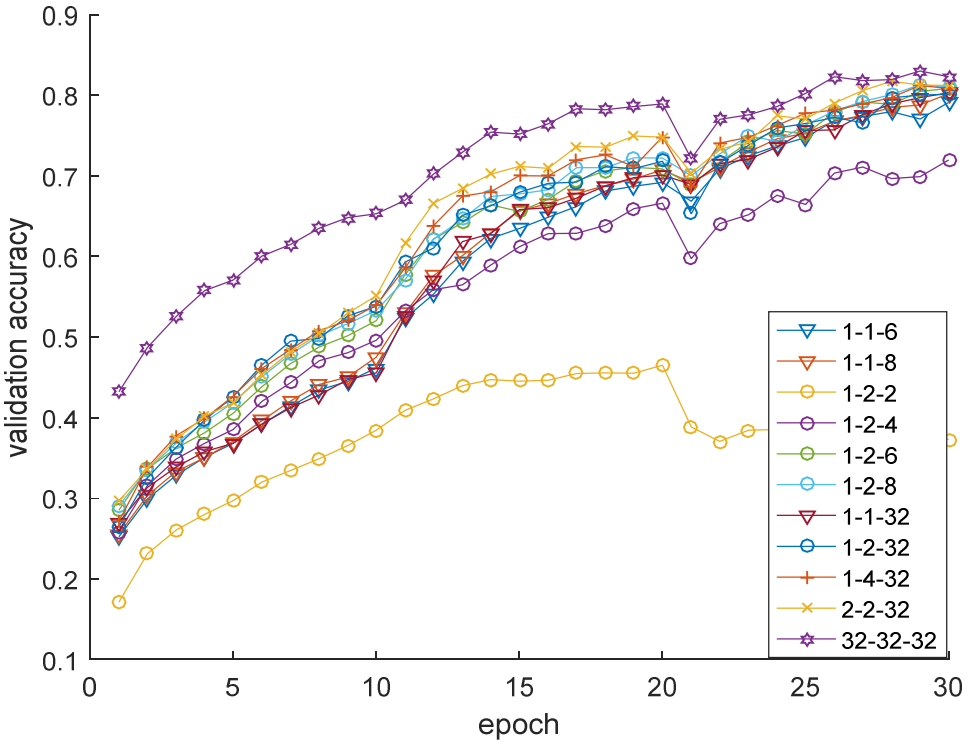


图 3-1 不同量化比特组合下 AlexNet 在 CIFAR-10 数据集上测试准确率对比
（图例解释：1-1-6 表示网络权值 1 比特，激活值 1 比特， 导数 6 比特）

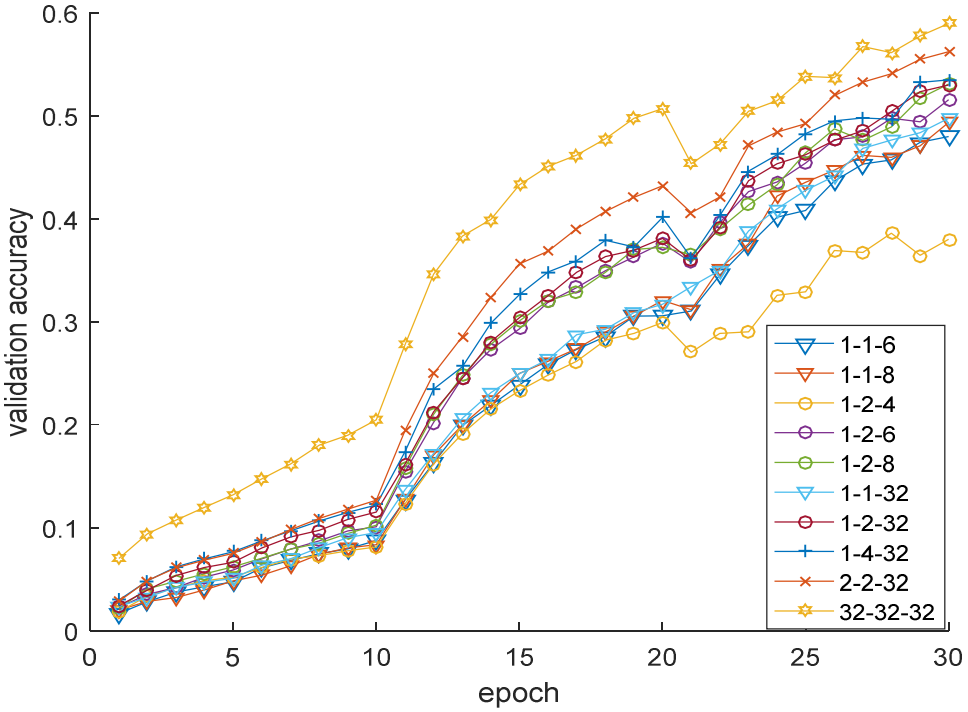


图 3-2 不同量化比特组合下 AlexNet 在 CIFAR-100 数据集上测试准确率对比
（图例解释：1-1-6 表示网络权值 1 比特，激活值 1 比特，导数 6 比特）

3.1.2 网络逐层输出特征可分性研究

混合比特量化网络量化比特数的变化应该按照什么规律进行变化，这是混合量化比特网络研究另外一个重要问题。为此，我们研究了网络逐层特征输出的特点。网络输出是高维向量，可视化网络输出需要采用降维方法。实验中采用两种降维方法，PCA[42]和 t-SNE[43]，研究对象是全精度 VGG-like 网络。实验结果如下：

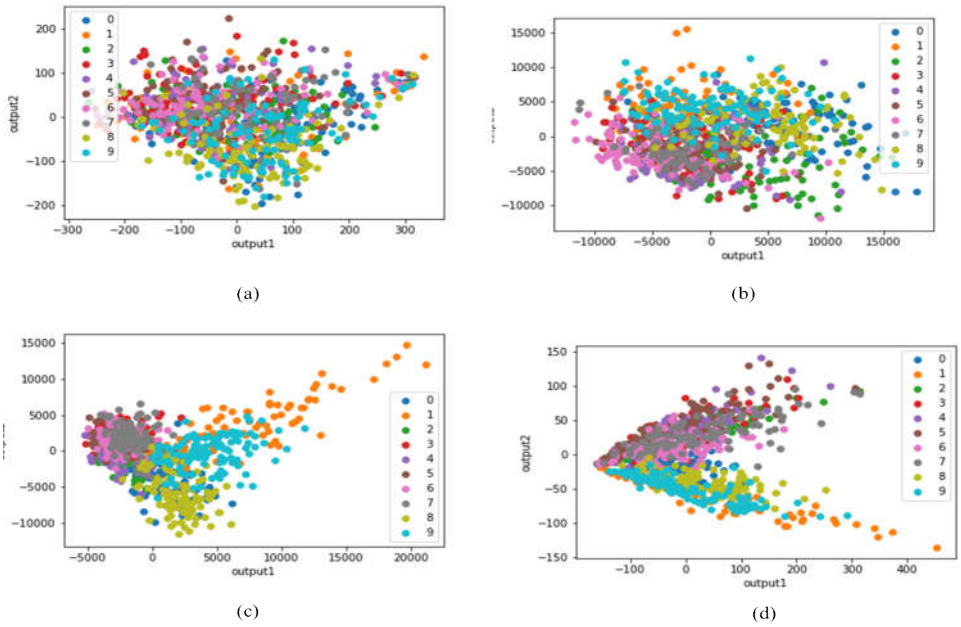


图 3-3 PCA 降维下 VGG-like 网络不同层不同类别特征可分性图
（（a）第 2 层 （b）第 8 层 （c）第 19 层 （d）第 21 层）

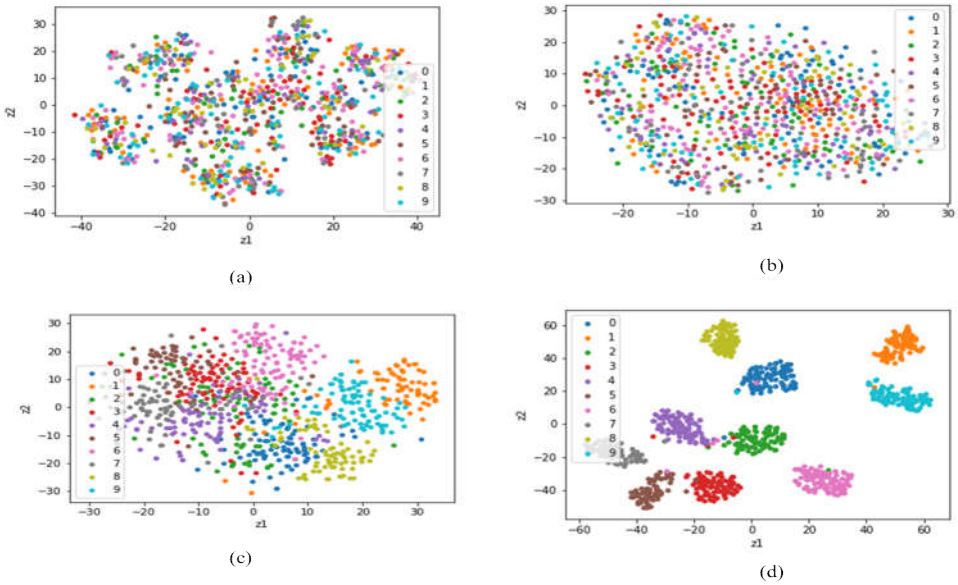


图 3-4 t-SNE 降维下 VGG-like 网络不同层不同类别特征可分性图
（（a）第 2 层 （b）第 8 层 （c）第 19 层 （d）第 21 层）

由于 t-SNE 降维算法适用于非线性降维，降维效果比 PCA 好。根据实验结果，我们可以发现随着量化的网络层数增加，量化网络输出特征可分性越来越好，相同类别的特征分布

越来越集中，不同类别中心之间距离越来越大。因此对于每层量化位数，我们提出了量化位数逐层递减的网络机制。对于可分性不好的层采用更多的量化比特，对于可分性好的层采用更少的量化比特。

3.2 任意比特量化方式

混合比特量化网络中含有多比特权值和激活值，因此本节主要讲述网络参数和激活值的多比特量化方式。由于在三值化网络的研究中，我们发现稀疏区间能够提高网络的性能，因此我们提出的多比特量化方式和旷视科技的 DoReFa-Net 网络的量化方式不同，我们的多比特量化方式带有稀疏区间。

网络中权值参数和激活值的量化方式采用的是 rounding 量化办法。对于网络中的量化前权值参数 w，其中取值范围为[-1,1]。量化函数如下：

$$quantize(w) = sign(w) * \frac{round(|w| * (2^{k-1} - 1))}{2^{k-1} - 1} \tag{3-1}$$

其中 sign(w)为对于 w 取符号函数，round()为取整函数。通过这个函数，给定量化比特 k，则量化的离散取值为 $\{-1, -\frac{2^{k-1}-2}{2^{k-1}-1}, -\frac{2^{k-1}-3}{2^{k-1}-1}, ..., -\frac{1}{2^{k-1}-1}, 0, \frac{1}{2^{k-1}-1}, ..., \frac{2^{k-1}-3}{2^{k-1}-1}, \frac{2^{k-1}-2}{2^{k-1}-1}, 1\}$，包含 2^k-1 个量化取值。量化取值包含 0，量化函数中稀疏区间宽度为 $\frac{1}{2^{k-1} - 1}$。

对于量化前网络激活值 a，先通过 Hardtanh 层将取值范围转换为[-1,1]，再通过 ReLU 层将范围限制在[0,1]。量化函数如下：

$$quantize(a) = \frac{ceil(a * (2^k - 1))}{2^k - 1} \tag{3-2}$$

其中 ceil()为向上取整函数，给定量化比特 k，则量化的离散取值为 $\{0, \frac{1}{2^k-1}, \frac{2}{2^k-1}, ..., \frac{2^k-3}{2^k-1}, \frac{2^k-2}{2^k-1}, 1\}$，包含 2^k 个取值。量化取值包含 0，但是只有当量化前网络激活值 a 为 0 的时候，量化后网络激活值 a 才为 0。

下图为任意比特量化方式下网络权值和激活值的量化函数。

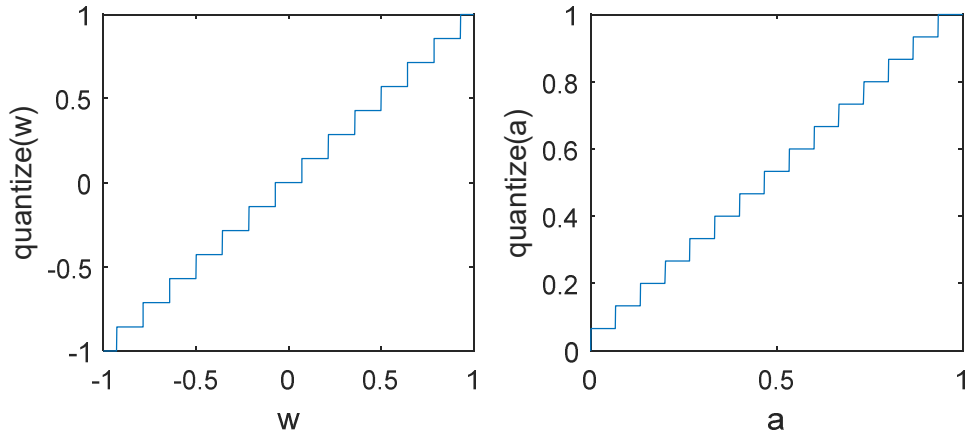


图 3-5　4 比特权值和激活值量化函数图

（左侧：权值参数量化 右侧：激活值量化）

3.3 网络结构和训练方法

混合量化网络的网络结构与其他量化网络类似，主要区别有两点：第一点是每一层量化比特设置，另外是需要在每个量化层后面添加自定义的 GrapMap 层。

关于每一层量化比特数的设置，主要有两点需要满足。第一点是根据网络逐层输出特征可分性的研究结论，前面层数的量化比特数目不小于后面层数的量化比特数目。第二点是网络的平均量化比特数目在 1 比特和 2 比特之间。表示成数学公式如下：

$$bit_1 \geq bit_2 \geq bit_3 \geq \ldots\ldots \geq bit_L \tag{3-3}$$

$$avg\_bit = \frac{\sum_{i=1}^{L} bit_i * params_i}{\sum_{i=1}^{L} params_i} \in (1,2)$$

对于本章实验部分测试的两种深度学习模型，VGG-like 网络的量化比特从 8 比特开始，每个卷积层或者线性层的量化比特除以 2。量化比特变为 1 时，后面层一直保持量化比特为 1 不变。而关于 ResNet 网络的量化比特数目，第一个卷积层为 8 比特，之后每个 BasicBlock 量化比特数目除以 2，当量化比特数变为 1 时，后续网络模块的量化比特数均为 1。

不同量化位数的比特层对于学习率的要求也是不一样的。如果将学习率理解为在梯度下降过程当中步长，量化比特数越多，每个量化值之间差值越小。如果采用过大的学习率的话，容易使得在梯度下降过程中跳过中间的量化值，造成网络的不稳定。量化比特越少，每个量化值之间差距越大，此时如果采用过小的学习率的话，容易使得到达全局最优过程缓慢。对于混合比特量化网络，需要设计根据量化位数调节学习率的方案。因此设计了 GrapMap 层，这一层正向和反向传播的过程如下：

Algorithm 2 GrapMap前向传播和反向求导实现

Require: 两个量化层之间的量化位数差距bitgap

Forward: $g_o = g_i$

Backward: $\frac{\partial L}{\partial g_i} = \frac{\partial L}{\partial g_o} * \frac{1}{2^{bitgap}}$

GrapMap 主要作用于反向求导过程，对于前面层来说，后面层的导数会乘以一个小于 1 的数，这个数和两个量化层量化位数差相关。改变导数的值和改变学习率的值本质上来说有相似之处，相当于前面层学习率减小，从而达到根据不同层量化比特数调节学习率的目的。

相对于第二章提及的二值化和三值化这两种每层等比特量化网络，混合比特量化网络的网络结构会有些变化。

图 3-6 逐层等比特网络和混合比特网络单元模块图
（左侧：逐层等比特网络　右侧：混合比特网络）

混合比特 VGG-like 网络结构和 ResNet 网络结构如下图所示：

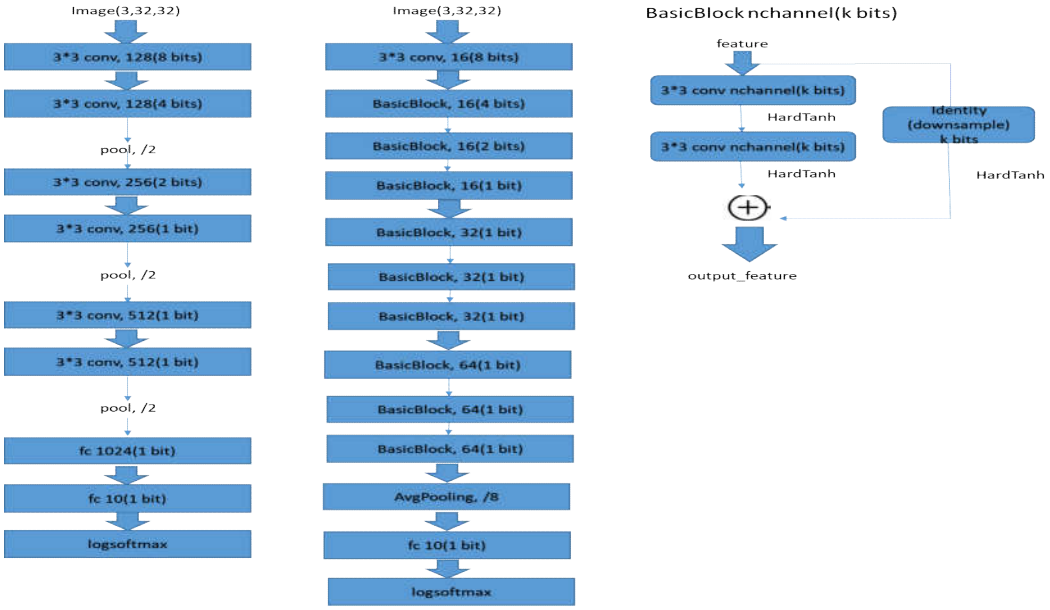


图 3-7 VGG-like 和 ResNet 混合比特网络比特分布图

混合比特网络量化函数反向求导过程和等比特网络基本一致，根据 STE（Straight Through Estimator）[33]的思想，损失函数对量化前的值的导数和损失函数对于量化后的值的导数相等。网络具体训练算法如 3-4 节下面的 Algorithm 3。

3.4 网络的比特运算

相较于二值化和三值化网络比特运算，混合比特量化网络比特运算推导更加容易，因为混合比特量化网络量化值直接可以用{0,1}序列表示。假设权值量化位数为 k 比特，激活值量化位数为 m 比特，则权值向量各个分量可以表示为：

$$w_i = \begin{cases} \dfrac{\sum_{l=0}^{k-2} c_l(w_i)2^l}{2^{k-1}-1} & c_{k-1}(w_i) = 0 \\ -\dfrac{\sum_{l=0}^{k-2} c_l(w_i)2^l}{2^{k-1}-1} & c_{k-1}(w_i) = 1 \end{cases} \tag{3-4}$$

而激活值向量各个分量可以表示为：

$$a_i = \frac{\sum_{l=0}^{m-1} c_l(x_i)2^l}{2^m-1} \tag{3-5}$$

假设权值向量 **w** 中分成两部分向量，第一部分向量包含为正数或者为 0 的分量，第二部分包含为负数的分量，记为 $\mathbf{w_P}$ 和 $\mathbf{w_N}$。对应激活值向量按照权值向量索引，分为两个部分，记为 $\mathbf{a_P}$ 和 $\mathbf{a_N}$。则量化后权值向量和激活值向量内积如下：

$$
\begin{aligned}
\mathbf{w}*\mathbf{a} &= \sum_{i=1}^{n} w_i a_i = \sum_{i=1}^{n_p} w_{pi} a_{pi} + \sum_{i=1}^{n_N} w_{Ni} a_{Ni} \\
&= \sum_{i=1}^{n_p} \left(\frac{\sum_{l=0}^{k-2} c_l(w_{pi}) 2^l}{2^{k-1}-1} \right) \left(\frac{\sum_{l=0}^{m-1} c_l(x_{pi}) 2^l}{2^m-1} \right) - \sum_{i=1}^{n_N} \left(\frac{\sum_{l=0}^{k-2} c_l(w_{Ni}) 2^l}{2^{k-1}-1} \right) \left(\frac{\sum_{l=0}^{m-1} c_l(x_{Ni}) 2^l}{2^m-1} \right) \\
&= \frac{1}{(2^{k-1}-1)(2^m-1)} \left(\sum_{l=0}^{k-2} \sum_{g=0}^{m-1} 2^{l+g} (\text{popcount}[\text{and}(c_l(\mathbf{w}_P), c_g(\mathbf{x}_P))] - \right. \\
&\left. \text{popcount}[\text{and}(c_l(\mathbf{w}_N), c_g(\mathbf{x}_N))] \right)
\end{aligned}
\tag{3-6}
$$

相对于二值化和三值化网络，多比特网络在比特运算上复杂度会更高，花费时间更多。混合比特网络只是在某些层使用多比特量化方式，大部分为单比特，因此相较于全部层使用多比特量化方式，运算时间和存储上会有很大提升。

Algorithm 3 混合比特量化网络算法

Require: 一个batch的输入数据和对应标签(a_0, a^*)，网络的比特向量bitmap，上次迭代后的网络权值参数W，上次迭代过程归一化层参数θ，上次迭代过程学习率η，学习率衰减参数λ

Ensure: 当前迭代更新后权值参数W^{t+1}，更新后归一化层参数θ^{t+1}，以及更新后的学习率η^{t+1}

1.前向传播

for $k = 1$ *to* L **do**
  $W_k^q \leftarrow \text{Quantize}(W_k, \text{bitmap}_k)$
  $s_k \leftarrow a_{k-1}^q W_k^q$
  $a_k \leftarrow \text{BatchNorm}(s_k, \theta_k)$
  if $k < L$ **then**
    $a_k^q \leftarrow \text{Quantize}(a_k, \text{bitmap}_k)$(包括ReLU层和量化函数)
  end if
end for

2.反向求导

已知最后一层输出a_L以及数据标签a^*，计算损失函数对于最后一层输出值的梯度$g_{a_L} = \frac{\partial Loss}{\partial a_L}$。

for $k = L$ *to* 1 **do**
  if $k < L$ **then**
    $g_{a_k}{}^q \leftarrow \frac{g_{a_k}^{grapmap}}{2^{bitmap_{k+1} - bitmap_k}}$(GrapMap层的反向求导)
    $g_{a_k} \leftarrow g_{a_k}{}^q 1_{|a_k|}$
  end if
  $(g_{s_k}, g_{\theta_k}) \leftarrow \text{BackBatchNorm}(g_{a_k}, s_k, \theta_k)$
  $g_{a_{k-1}^q} \leftarrow g_{s_k} W_k{}^q$
  $g_{W_k^q} \leftarrow g_{s_k}{}^T a_{k-1}^q$
end for

3.参数更新

for $k = 1$ *to* L **do**
  $\theta_k^{t+1} \leftarrow \text{Update}(\theta_k, \eta, g_{W_k^b})$
  $W_k^{t+1} \leftarrow \text{Clip}(\text{Update}(W_k, \eta, g_{W_k^b}), -1, 1)$
  $\eta^{t+1} \leftarrow \lambda \eta$
end for

3.5 实验及结果

3.5.1 准确率对比

本节实验利用 VGG-like 和 ResNet-20 网络框架，在 CIFAR-10 和 CIFAR-100 数据集上，

对比二值化、三值化以及混合比特量化这三种结构下网络准确率变化。混合比特量化 VGG-like 和 ResNet 混合比特网络的比特分布已经在 3.3 节提及。训练集为 50000 张图片，测试集为 10000 张图片。batch size 是 100，优化方法均为 Adam，weight decay 为 0。学习率采用的是第二章实验中提到的 warm up+eponential decay[46]的方式，其中二值化网络最大学习率为 0.03，全精度、三值化和混合量化比特网络最大学习率为 0.02。

下面是这三种量化结构下 VGG-like 网络和 ResNet-20 网络在 CIFAR-10 数据集上的达到最高准确率。（后文实验中混合比特网络简称 8-4-2-1 网络）

表 3-1 量化及全精度 VGG-like 和 ResNet-20 网络在 CIFAR-10 数据集上达到准确率

	VGG-like	ResNet-20(inflate=4)
Float	0.9353	0.9340
Binary	0.8987	0.8864
Ternary	0.9156	0.9027
8-4-2-1	0.9169	0.9088

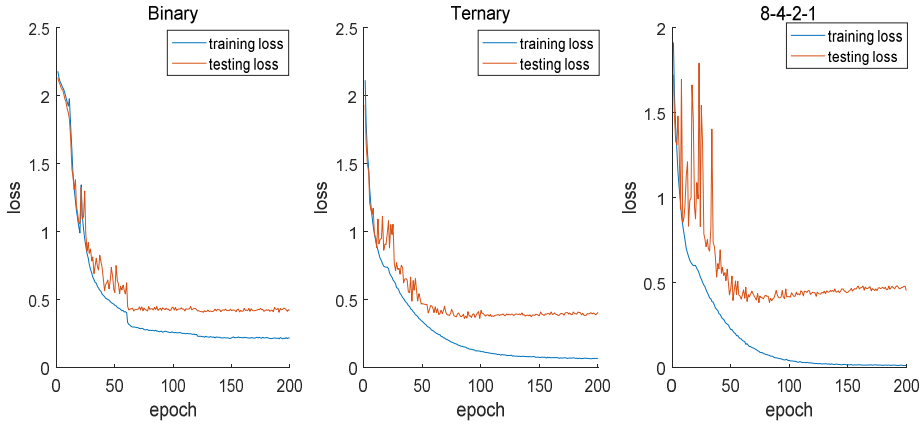


图 3-8 二值化、三值化以及混合比特 ResNet-20 网络在 CIFAR-10 数据集上 loss 曲线 ((1) 二值化网络 (2) 三值化网络 (3)混合比特网络)

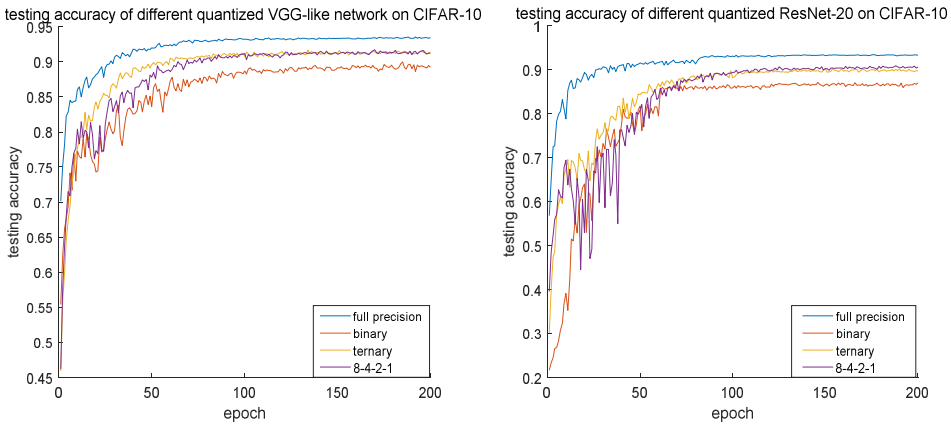


图 3-9 全精度和三种量化 VGG-like 网络和 ResNet-20 网络在 CIFAR-10 准确率曲线 （左侧：VGG-like 网络 右侧：ResNet-20 网络）

我们在 CIFAR-100 上进行同样的对比实验，和 CIFAR-10 数据集相比，CIFAR-100 数据

集类别数更多，分类难度也更大。所采用的参数和在 CIFAR-10 实验中所用到的数据一致。

表 3-2 量化及全精度 VGG-like 和 ResNet-20 网络在 CIFAR-100 数据集上达到准确率

	VGG-like	ResNet-20(inflate=4)
Float	0.6948	0.7469
Binary	0.6558	0.5973
Ternary	0.7121	0.6317
8-4-2-1	0.6929	0.6578

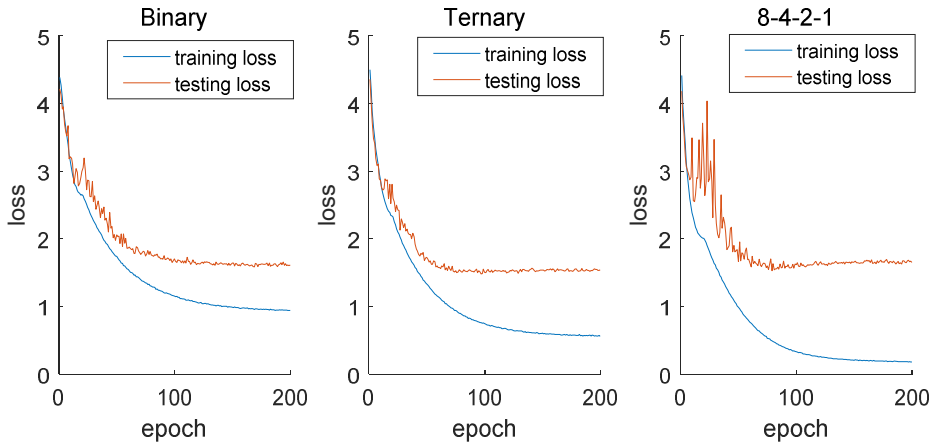


图 3-10 二值化、三值化以及混合比特 ResNet-20 网络在 CIFAR-100 数据集上 loss 曲线
((1) 二值化网络 (2) 三值化网络 (3)混合比特网络)

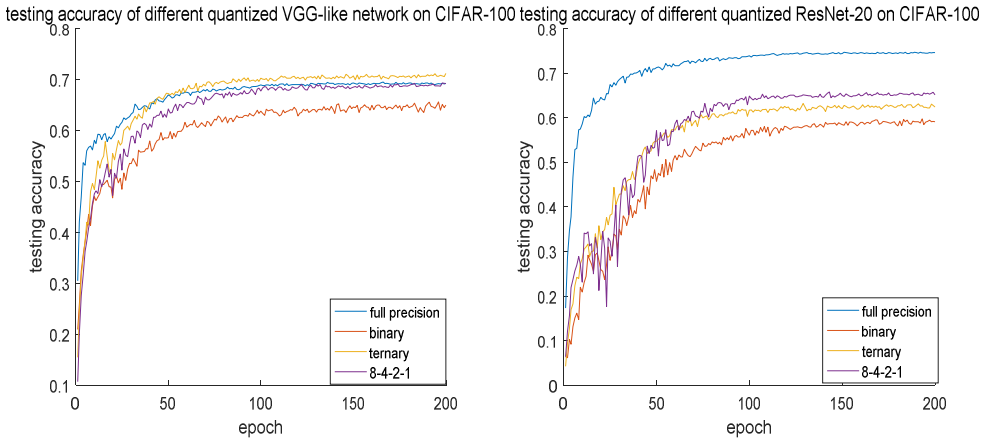


图 3-11 全精度和三种量化 VGG-like 网络和 ResNet-20 网络在 CIFAR-100 准确率曲线
（左侧：VGG-like 网络 右侧：ResNet-20 网络）

可以看到，在 CIFAR-10 和 CIFAR-100 数据集上，量化的 VGG-like 和 ResNet 网络带来准确率上损失很小，结果与全精度网络有 2-3%的距离。我们提出的混合比特量化结构在这两个数据集上的准确率远远超过二值化网络，并且逼近三值化网络的准确率结果。表 3-2 中在 CIFAR-100 数据集上出现三值化网络的准确率高于全精度网络的情况，这主要是全精度网络在 CIFAR-100 数据集上出现过拟合情况，而三值量化网络有一定稀疏区间，从而减轻网络在该数据集上过拟合程度。下面作出的是二值化、三值化以及混合比特量化网络中间层

特征输出图。相较于二值化和三值化网络，混合量化比特网络在第一层所用的比特数目较多，能够提取更多色彩信息，而后面层数即使使用较少比特数，不同类别特征输出可分性也更强。

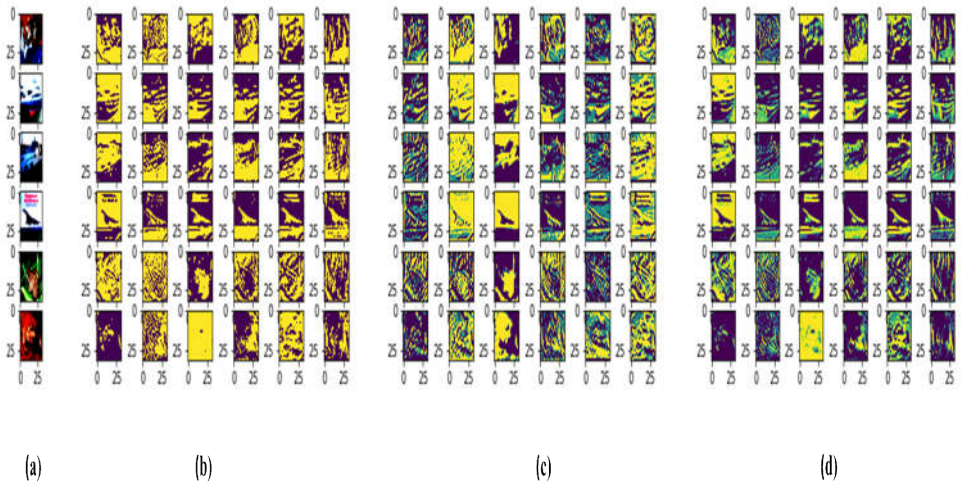


图 3-12 CIFAR-10 数据上二值化、三值化以及混合比特 VGG-like 网络第一层特征输出
((a)RGB 图片 (b)二值化网络 (c)三值化网络 (d)混合比特网络)

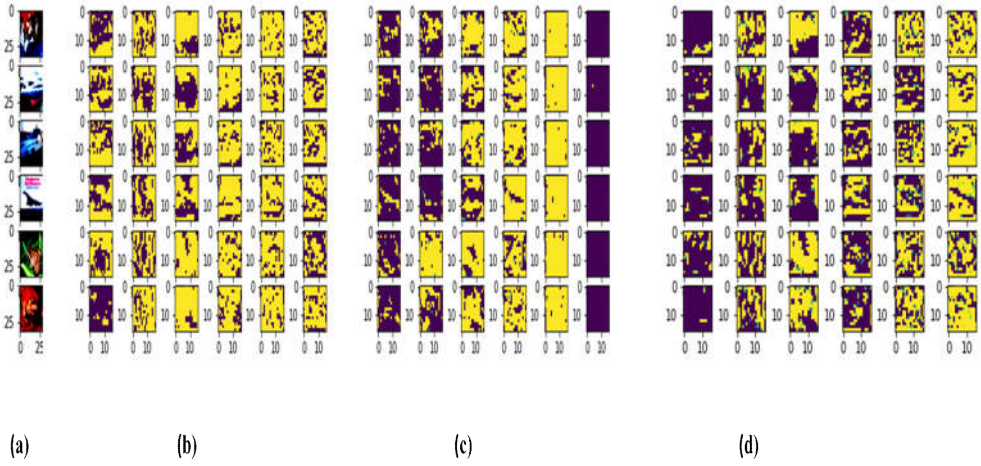


图 3-13 CIFAR-10 数据上二值化、三值化以及混合比特 VGG-like 网络第七层特征输出
((a)RGB 图片 (b)二值化网络 (c)三值化网络 (d)混合比特网络)

3.5.2 网络平均量化比特理论计算

量化网络量化后的大小和具体运算时间是网络部署主要内容，最后一小节不从部署角度来分析混合比特网络的压缩率，而是从平均量化比特角度分析混合量化 VGG-like 网络和 ResNet-20 网络压缩大小。

先分析 VGG-like 网络，其每层参数数量分布如下：

表 3-3 VGG-like 网络卷积层和线性层参数个数及量化比特

	Number of Parameters	Quantization Bits
Conv1	3*3*3*128=3456	8
Conv2	128*3*3*128=147456	4
Conv3	128*3*3*256=294912	2
Other Layer	12527616	1

混合比特 VGG-like 网络平均比特数为：

$$\frac{3456*8+147456*4+294912*2+12527616}{3456+147452+294912+12527616}=1.06 \quad (3\text{-}7)$$

最终量化比特为 1.06 比特，网络压缩率为原来的体积的 $\frac{1}{30}$

接下来看下 ResNet-20 网络的平均量化比特数目，与 VGG-like 网络逐个卷积层量化位数减小不同，ResNet-20 网络是每一个 BasicBlock 进行量化位数减小的。

表 3-4 ResNet-20 网络各层参数个数及量化比特

	Number of Parameters	Quantization Bits
Conv1	3*3*3*16=432	8
BasicBlock1	16*64*3*3*2=18432	4
BasicBlock2	64*64*3*3*2=73728	2
Other Layer	331776+663552+2560=997888	1

混合比特 ResNet-20 网络的平均比特数为：

$$\frac{432*8+18432*4+73728*2+997888*1}{432+18432+73728+997888}=1.12 \quad (3\text{-}8)$$

最终量化比特为 1.12 比特，网络压缩率为原来体积的 $\frac{1}{28}$ 。

3.5.3 模型体积和运算时间对比

上一节计算的是CIFAR-10数据集上混合比特量化网络所用的平均量化比特及理论的压缩率，我们也测试了二值化、三值化以及混合比特量化网络的实际模型体积以及在测试集10000张图片上所用的运算时间。

表 3-5 CIFAR-10 数据集上量化及全精度 VGG-like 网络体积大小及压缩率对比

	After Compression (MB)	*Actual Compression Rate*	*Ideal Compression Rate*
Float	51.90	x1.0	x1
Binary	1.70	x30.5	x32
Ternary	3.55	x14.6	x16
8-4-2-1	1.81	x28.7	x30

表 3-6 CIFAR-10 数据集上量化及全精度 VGG-like 网络测试运算时间对比

	Time Consumption (s)	*Actual Acceleration Rate*	*Ideal Acceleration Rate*
Float	7.300	x1	x1
Binary	0.146	x50	x58
Ternary	0.243	x30	x36
8-4-2-1	0.152	x48	x52

表 3-7 CIFAR-10 数据集上量化及全精度 ResNet-20 网络体积大小及压缩率对比

	After Compression (MB)	*Actual Compression Rate*	*Ideal Compression Rate*
Float	17.4	x1.0	x1
Binary	0.56	x30.9	x32
Ternary	1.14	x15.2	x16
8-4-2-1	0.65	x26.5	x28

表 3-8 CIFAR-10 数据集上量化及全精度 ResNet-20 网络测试运算时间对比

	Time Consumption (s)	*Actual Acceleration Rate*	*Ideal Acceleration Rate*
Float	5.600	x1	x1
Binary	0.107	x52	x58
Ternary	0.181	x31	x36
8-4-2-1	0.119	x47	x50

根据上面已有数据，可以作出二值化、三值化以及混合比特量化的 VGG-like、ResNet-20 在测试准确率、模型压缩率及加速率的三维性能散点图。

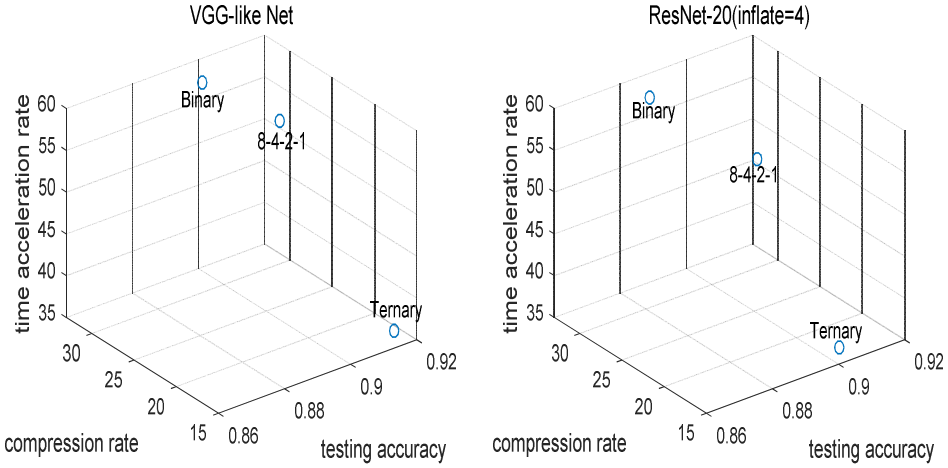


图 3-14 三种量化网络测试准确度、模型压缩率及运算加速率特性图

可以看到，我们提出的混合比特量化网络仅仅牺牲很小压缩率及运算加速率的基础上能够实现测试准确率一定程度的提升，说明这种量化网络结构能够在增加少量比特情况下达到更优的准确率结果。

3.6 本章小结

本章提出了和第二章结构有所区别的混合比特量化网络结构，首先通过两个实验引出了我们提出混合比特量化网络构想的缘由和理论基础。多比特量化网络实验解释了在网络中使用多比特权值和激活值能够提高量化网络的准确率，而网络每层输出特征可分性实验解释混合比特网络结构中量化比特为什么从前往后逐渐递减。接下来分别提出了采用的多比特量化方式、混合量化网络结构与训练方法。多比特量化方式使用的是包含稀疏区间的多比特量化方式，训练方法的不同点是加入 GrapMap 层，使得不同量化比特层在参数更新时学习率不同。最后通过实验，以 VGG-like 网络和 ResNet-20 为例，计算得到的混合量化网络平均比特数在 1-2 比特之间。在 CIFAR-10 和 CIFAR-100 数据集上对比其与二值化和三值化网络的测试准确率、网络体积和运算时间。网络体积和运算时间增加很小的情况下，混合比特网络的测试准确率远远超过二值化网络，逼近甚至超越三值化网络。

第四章 量化网络与知识蒸馏

第二章提到量化网络的正则化方法，核心思想是将先验知识加入网络训练过程之中，如量化网络正则化基于在使得量化误差尽可能小的情况下使得更多量化前的权值参数趋向于0，量化后网络更加稀疏的先验知识。而正则化是否有用，很重要的是我们的先验假设是否合理。而相对于全精度网络，量化网络将量化过程加入训练过程之中，损失函数中已经包含最小化量化误差，因此我们的正则化项的效果不一定能够体现。实验结果表明量化网络中正则化项带来权值分布不是网络性能最佳时的权值分布。因此一种新的想法很自然产生，是否可以给量化网络更多提示信息，而这种提示信息不是我们人为设计，而是由一个学习能力更强的网络提供的。训练小型网络里面的一种新型方法知识蒸馏[23-25]，和量化网络就联系起来了。

4.1 单热向量与 Softmax 输出

知识蒸馏方法涉及到多分类中两种类型的向量，一种是单热向量，另外一种是 Softmax 输出向量。对于一个多分类问题，假设类别总数为 k，单热向量和实值输出向量都是 k 维。单热向量指的是在样本的类别那一个维度上数值为 1，其他维度上数值为 0 的向量。而 Softmax 输出[44]指的是对一个网络最后线性层的 k 维输出向量 **x** 的每个分量作如下变换：

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_i}} \qquad i = 1, 2 \ldots . k \tag{4-1}$$

Softmax 相当于把线性层输出向量由原来分布转化为在区间[0,1]中的概率分布。相对于原有的单热向量来说，Softmax 输出的信息熵更大，所含的信息量也更多。其中不仅包含单热向量所拥有的属于哪一类的绝对信息，而且包含属于不同类别的概率程度差异的相对信息。

$$\begin{aligned} H(one-hot) &= 0 \\ H(Softmax) &= -\sum_{i=1}^{k} \frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}} \log\left(\frac{e^{x_i}}{\sum_{j=1}^{k} e^{x_j}}\right) > 0 \end{aligned} \tag{4-2}$$

因此使用 Softmax 输出或者 Softmax 变形输出作为网络的 Soft Target，比原来单纯使用单热向量作为 Hard Target，将使得网络取得更好泛化性能。对于小网络而言，小网络学习能力不足，如果给予的目标信息不够，容易使得网络在训练集上产生过拟合现象。更加充足的目标信息，使得小网络参数有更多调整空间和方向，从而使得网络在测试集上取得效果更优。

4.2 知识蒸馏方法

Soft Target 在网络泛化方面具有重要意义，知识蒸馏方法结合 Soft Target 和 Hard Target 的特点。Hard Target 来源于样本标签对应的单热向量，而 Soft Target 来源于 teacher 网络的最后一层输出。在知识蒸馏中，teacher 网络指的是在训练和测试数据集上表现更好，复杂程度更高的网络。对应的，student 网络指的是学习能力不足，复杂度比较低的网络。Soft Target 和 Softmax 输出定义不完全一致，Soft Target 是在原有最后一层线性层输出基础上，除以一个温度系数 T，再进行 Softmax 运算得到的，表达式如下：

$$q_i = \frac{e^{z_i / T}}{\sum_{j=1}^{k} e^{z_j / T}} \tag{4-3}$$

当 T=1 时，Soft Target 和原有的 Softmax 输出保持一致。温度系数 T 越大，在相同的线性层输出下，得到的在[0,1]之间的概率分布更加平缓。下图所示为给定 10 个线性层输出样本，在不同的温度系数 T 下，对应 10 个 Soft Target 向量的热度图分布情况。随着温度系数增大，Soft Target 向量的各分量分布越来越平缓，差异越来越小。适当温度系数能够让 Soft Target 使网络获得更好泛化能力，但过大温度系数将会掩盖掉样本属于哪类这个重要信息并且会引入错误信息。

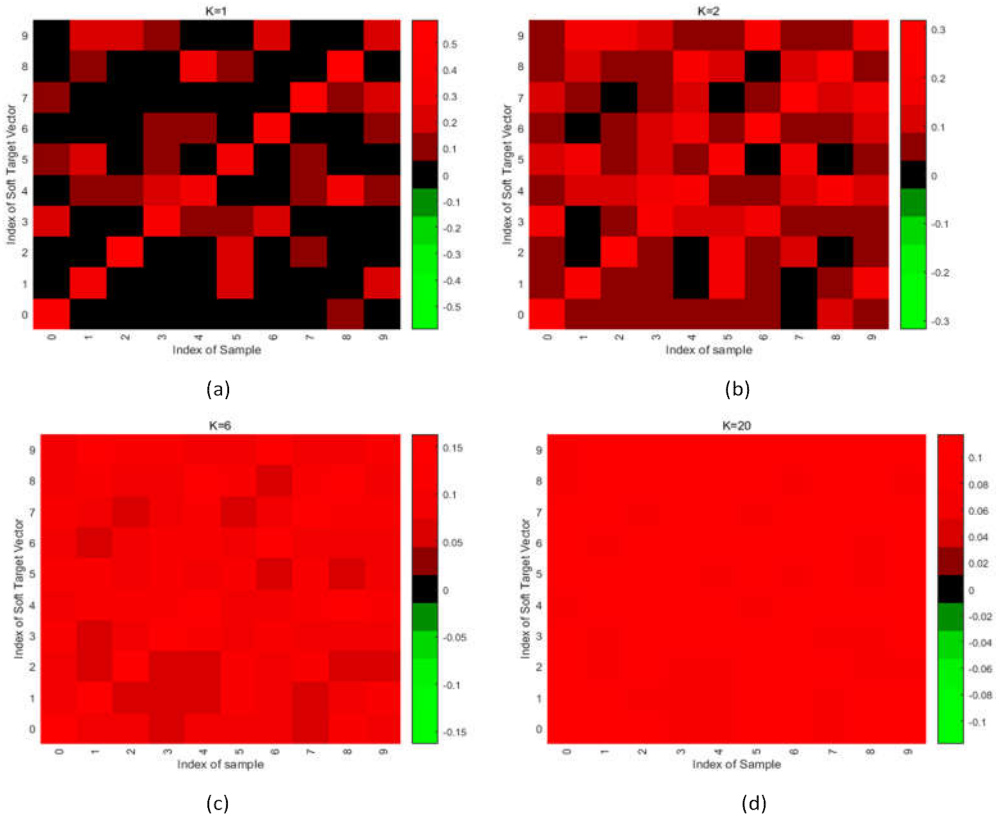


图 4-1 不同温度系数对于 Soft Target 中类别差异度影响

((a) K=1 (b)K=2 (c)K=6 (d)K=20)

student 网络训练过程中损失函数包含两个部分，第一个是 student 网络 Soft Target 和 teacher 网络 Soft Target 之间的 KL 散度（Kullback-Leibler divergence），其衡量两个 Soft Target 的信息熵的差值。第二个是 student 网络最后 Softmax 输出和样本 Hard Target 之间的交叉熵函数，损失函数公式如下：

$$L_{KD} = \alpha T^2 KL\_divergence(Q_S^{soft}, Q_T^{soft}) + (1-\alpha)CrossEntropy(Q_S, label) \quad (4\text{-}4)$$

其中 α 是比例系数，起到的是类似正则化系数的作用。由于为了弥补因为温度系数带来的两部分损失函数之间的差距，KL 散度函数前需要乘上 T 的平方。下图为知识蒸馏下学生网络整个训练和测试过程：

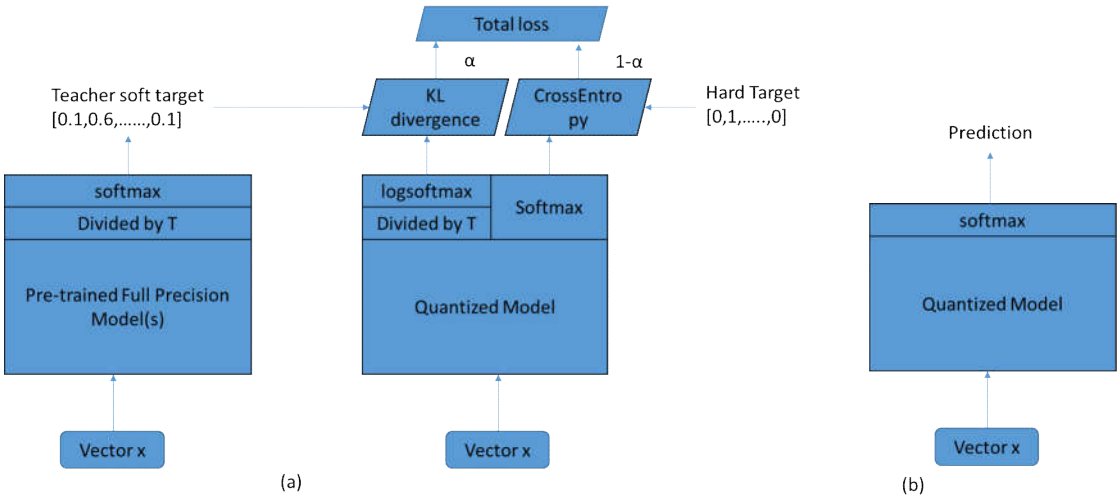


图 4-2 知识蒸馏下量化网络训练和测试过程示意图

((a) 训练过程 (b)测试过程)

在整个训练过程中，给定输入向量 **x**，对于全精度的 teacher 模型，计算 Soft Target 输出。而对于量化的 student 模型，前向传播得到最后线性层输出，之后一方面根据公式(4-3)计算 Soft Target 输出，另一方面根据公式(4-1)计算出正常的 Softmax 输出。通过公式(4-4)计算出损失函数值，在反向传播过程中，只对于 student 模型中的参数计算导数，对于 teacher 模型不进行反向求导和参数更新过程。在测试过程，student 模型前向传播计算出线性层输出，经过 Softmax 后得到向量的最大值分量，即为预测类别值。

4.3 实验及结果

4.3.1 准确率对比实验

本节实验主要研究在知识蒸馏作用下，量化网络的准确率是否能够提升，以及在不同温度系数 T 和正则化系数 α 作用下，量化网络准确率提升情况。实验中使用的数据集为 CIFAR-10 数据集，训练集和测试集划分与之前章节相同。实验中使用的基本网络为 ResNet 网络，其中 teacher 网络是全精度 ResNet-30(inflate=4)网络，而 student 网络有三种，按照学习能力从强到弱顺序，分别是混合比特 ResNet-20(inflate=4)、二值化 ResNet-20(inflate=4)以及混合比特 ResNet-20(inflate=1)。在量化的 ResNet 网络中，网络宽度系数(inflate)是一个很重要的因素，网络过窄将会使得量化网络学习能力变得很弱。下图展现实验中 teacher 和 student 网络在 CIFAR-10 上的测试准确率情况，其中 teacher 网络(全精度 ResNet-30)的测试准确率为 0.9340，上面提到三个 student 网络的测试准确率在没有 teacher 网络协助情况下测试准确率为 0.9088，0.8802 和 0.8165。

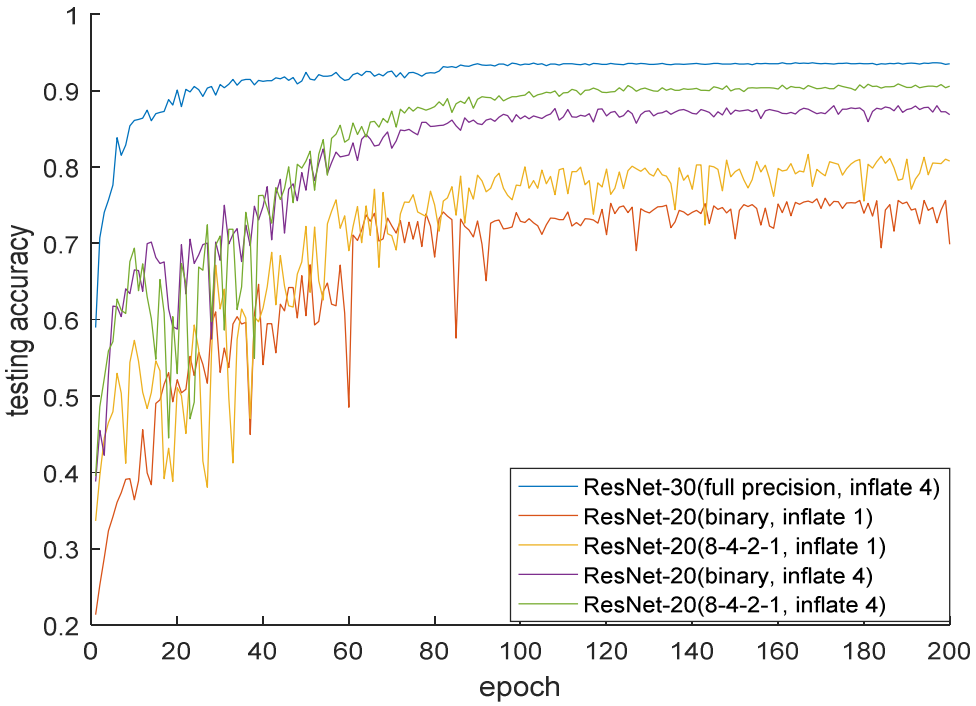


图 4-3 实验中用到 teacher 和 student 的 ResNet 网络在 CIFAR-10 数据集上测试准确率

对于同一个 student 网络的系列实验，实验参数设置是一致的。其中 batch size 为 100，优化算法为 Adam，weight decay 为 0，学习率采用前面章节 warm_up+exponential decay 方式，其中二值化网络最大学习率为比特网络最大学习率为 2e-2。首先研究了知识蒸馏参数对于训练 student 网络准确率影响，通过网格调参法选取能够提升 student 量化网络准确率最多的参数组合。下图为固定正则化系数，变化温度系数以及固定温度系数，变化正则化系数时混合比特 ResNet-20（inflate=4）的网络准确率变化。

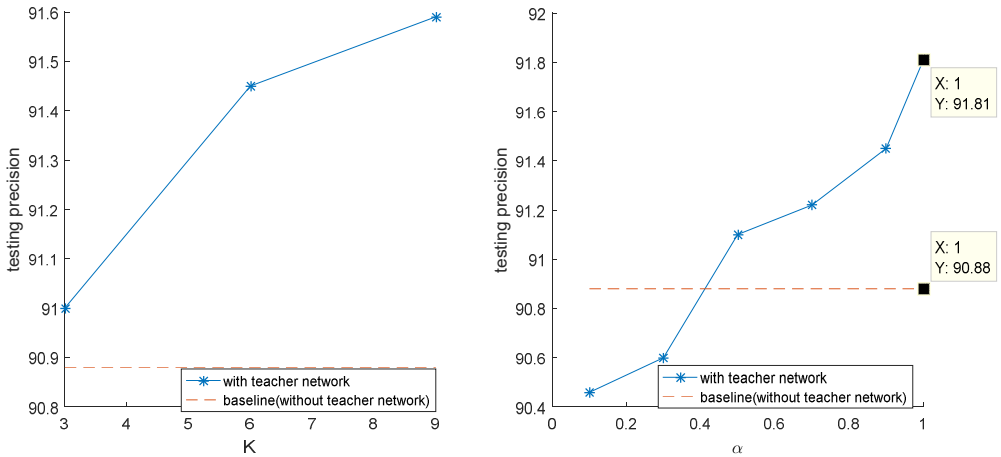


图 4-4 知识蒸馏中温度系数 K 和正则化系数 α 对于网络性能的影响
（左侧：与 K 的关系（α=0.9）， 右侧：与 α 的关系（K=6））

我们可以看到，使用知识蒸馏方法后，当正则化系数 α 取 1.0，温度系数 K=6 时，即完全使用 Soft Target，而不适用 Hard Target 时，混合比特 ResNet-20(inflate=4)在 CIFAR-10 上测试集准确率可以提升将近 1%。对于其他学习能力稍微弱点的 student 量化网络也进行研

究，发现对于二值化 ResNet-20(inflate=4)的量化网络准确率提升为 1.3%，对于混合比特 ResNet-20(inflate=1)却只有 0.2%， 说明知识蒸馏对于量化网络中学习能力较强的网络性能有一定提升效果，学习能力较弱的网络性能提升不明显。下图为三种类型量化的 student 网络在有 teacher 网络和没 teacher 网络情况下测试准确率曲线。

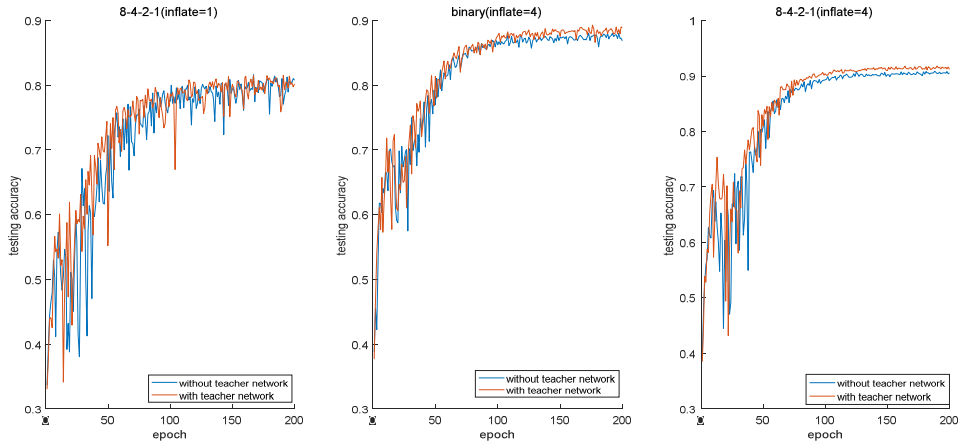


图 4-5 知识蒸馏下三种不同学习能力的量化 student 网络测试准确率曲线
（（1）8-4-2-1（inflate=1）， （2）binary（inflate=4） （3）8-4-2-1（inflate=4））

4.3.2 类别信息实验

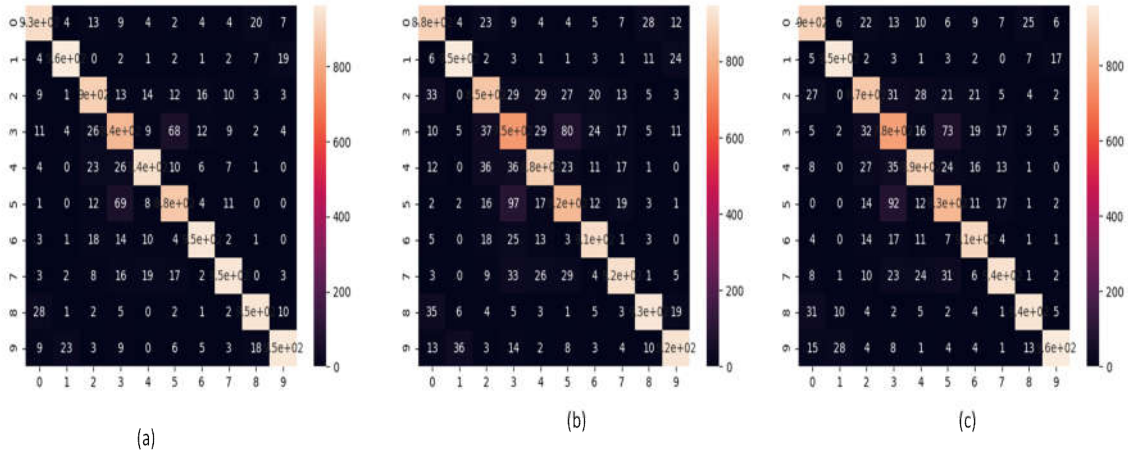
在量化的 student 网络训练过程中，teacher 网络的 Soft Target 中的传入信息有没有被 student 网络真正学习到，是知识蒸馏中一个重要问题。对于这个问题，我们可以采用类别信息分析方法进行检验。teacher 网络的 Soft Target 输出包含信息对于 student 网络在不同类别识别准确率上提升程度是不一样的，包含信息可能会使 student 网络在某些类别识别率取得较大提升，某些类别识别率提升效果不是很好。我们可以统计全精度 teacher 网络、没有 teacher 网络协助的量化网络以及有 teacher 协助的量化网络在各个类别上分类正确和分类错误样本数目。对于全精度 teacher 网络，可以计算全精度在各个类别上带来准确率提升率。假设总共有 k 个类别，计算公式如下：

$$r_{full}(i)=\frac{correct_{full}(i)-correct_{no\_teacher}(i)}{correct_{no\_teacher}(i)}\quad i=0,1,2....k-1 \tag{4-5}$$

同样对于有 teacher 的量化网络，相比没有 teacher 的量化网络在各个类别上的准确率也有一定提高，计算公式和(4-5)类似。

$$r_{dis}(i)=\frac{correct_{with\_teacher}(i)-correct_{no\_teacher}(i)}{correct_{no\_teacher}(i)}\quad i=0,1,2....k-1 \tag{4-6}$$

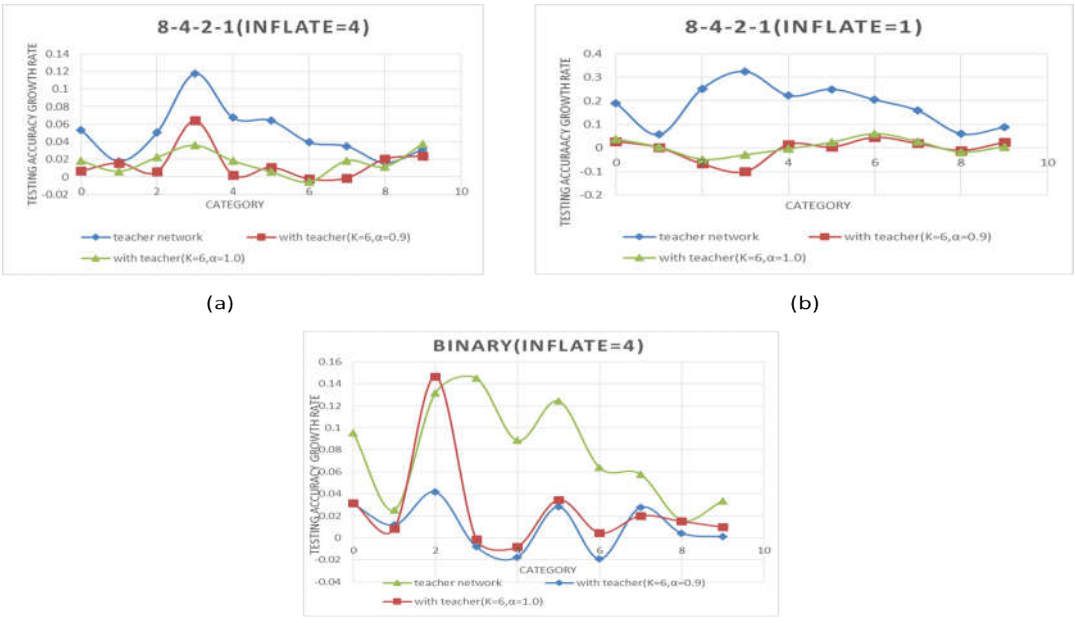
每个类别的分类正确样本数目可以通过混淆矩阵(confusion matrix)获得，其中矩阵对角线元素即为样本预测值和标签值相同元素个数。对于全精度 ResNet-30(inflate=4)、没有 teacher 网络协助的混合比特 ResNet-20(inflate=4)以及有 teacher 网络协助的混合比特 ResNet-20(inflate=4)，其混淆矩阵如下图所示：



(a) (b) (c)

图 4-6 teacher 网络、without teacher 和 with teacher 的 student 网络的混淆矩阵
((a) teacher 网络 (b) without teacher 的 student 网络 (c)with teacher 的 student 网络)

对于本章实验中的三个 student 网络，根据(4-5)和(4-6)计算全精度网络相对 student 量化网络在各个类别上准确率提升率以及加入 teacher 网络后对于 student 量化网络在各个类别上准确率提升率。绘制提升率随类别变化曲线，如下图所示：



(a) (b) (c)

图 4-7 不同学习能力 student 量化网络和 teacher 网络不同类别识别率增长曲线
((a) 8-4-2-1(inflate=4) (b)8-4-2-1(inflate=1) (c) binary(inflate=4))

根据前面实验结果，这三种 student 网络中，混合比特 ResNet-20(inflate=4)学习能力最强，而混合比特 inflate=1 的 ResNet-20 学习能力最弱。我们可以看到，由于量化网络相对而言学习能力较弱，有 teacher 网络协助时量化 student 网络的类别准确率增长率相比全精度的 teacher 的而言，有所下降，这说明 teacher 网络输入的信息在量化 student 网络学习过程中有所削弱。但也可以看到，对于混合比特 ResNet-20(inflate=4)而言，其类别准确率增长率最大值出现在第 3 类，和全精度 teacher 网络的类别准确率增长率最大值位置是一致的，并且曲线形状相似，说明当量化网络具有一定学习能力时，其能够学到 teacher 网络提供信息，并

且利用信息优化网络，在测试集上取得更好准确率。而对于学习能力较弱的网络(混合比特ResNet-20(inflate=1))，其类别准确率增长曲线和 teacher 网络基本不一致，说明知识蒸馏方法对于学习能力很弱的量化网络类型没有太好准确率提升作用。

根据本节实验讨论，知识蒸馏对于量化网络泛化性能提升具有一定作用，能够帮助量化网络在测试集上准确率取得提升。但其中很重要的因素是量化网络自身学习能力，对于部分学习能力极弱的量化网络，蒸馏的信息无法被网络很好的吸收，网络性能提升也比较困难。对于如何利用知识蒸馏使得部分学习能力很弱的量化网络（网络宽度很窄）的性能提升，需要后续继续研究。

4.4 本章小结

本章延续第二章正则化部分，在不增加量化比特基础上，在网络训练过程中添加更多信息，提升网络测试准确率。本章研究的知识蒸馏方法利用已经训练好的全精度网络，将其线性层输出除以温度系数，得到的 Soft Target 加入量化网络训练的损失函数中，为量化网络训练提供更多类别信息。前两节介绍了 Soft Target 相对于单热标签向量 Hard Target 的优势以及知识蒸馏的具体过程。实验部分使用三种学习能力不同的量化网络作为 student 网络，将全精度的 teacher 网络加入训练，通过准确率对比和不同类别识别准确率的提高率的对比，发现对于学习能力较强的量化网络，蒸馏的知识在一定程度上可以被吸收，并且使其在 CIFAR-10 数据集上测试准确率提升 1%左右。而对于学习能力较弱的量化网络，蒸馏知识难以被吸收，知识蒸馏方法不能起到很好作用。

第五章 总结与展望

本文主要研究了二值化、三值化以及混合比特量化这三种网络搭建和训练，以及如何在压缩网络权值参数在 1-2 个比特情况下提升测试准确率的方法。首先从量化方式、训练方法和前向比特运算三个方面对于二值化、三值化网络进行阐述，对比其与全精度网络测试准确率、网络体积以及运算时间三个方面的性能。其次，根据二值化和三值化网络思路，提出混合比特量化网络结构。平均量化比特在 1 至 2 比特之间的情况下，在 CIFAR-10 和 CIFAR-100 数据集上的准确率可以逼近甚至超越两比特网络。通过对于测试准确率、网络压缩率以及运算时间加速分析，可以得出混合比特量化网络能够实现在牺牲很小的网络压缩率和运算时间加速下，一定程度提高测试准确率。最后，提出利用正则化和知识蒸馏的方法，在量化网络训练过程中加入先验知识，在不增加网络量化比特的情况下提升网络的测试准确率。由于我们对于网络最佳权值参数分布的先验知识把握不准确，正则化方法对量化网络性能效果提升不佳。而将全精度网络蒸馏获得的知识加入量化网络训练过程的知识蒸馏方法，在量化网络学习能力足够情况下，在 CIFAR-10 数据集上能够提升测试准确率将近 1%。

毕业设计工作只是量化网络领域很小的一部分。对于未来工作，我们会在更多数据集上，包括现在跑的 ImageNet 数据集上，测试混合比特量化网络性能。对量化方式进行更多改进，争取能够超越最新发表论文中的 benchmark。此外，对于知识蒸馏方法，现有还是使用的是原有对小型化的全精度网络的训练方法，没有加入过多量化网络自身的特点，我们将会在知识蒸馏方法中加入量化特点，使得知识蒸馏方法能够帮助学习能力较弱的量化网络提高测试准确率。最后的话，我们期望能够与微纳电子系合作，将量化网络在 FPGA 等嵌入式平台上实现部署，使得量化网络的思想真正落到实处，用到产品线上，相信这将是一个艰巨的挑战。

参考文献

[1] He K, Zhang X, Ren S, et al.: Deep Residual Learning for Image Recognition, 2016 Ieee Conference on Computer Vision and Pattern Recognition, 2016: 770-778.

[2] Krizhevsky A, Sutskever I, Hinton G E. ImageNet Classification with Deep Convolutional Neural Networks[J]. Communications of the Acm, 2017, 60(6): 84-90.

[3] Williams R J, Zipser D. A learning algorithm for continually running fully recurrent neural networks[J]. Neural Computation, 1989, 1(2): 270-80.

[4] Goodfellow I J, Pouget-Abadie J, Mirza M, et al.: Generative Adversarial Nets, Ghahramani Z, Welling M, Cortes C, Lawrence N D, Weinberger K Q, editor, Advances in Neural Information Processing Systems 27, 2014.

[5] Girshick R, Donahue J, Darrell T, et al.: Rich feature hierarchies for accurate object detection and semantic segmentation, 2014 Ieee Conference on Computer Vision and Pattern Recognition, 2014: 580-587.

[6] Qin Y, Zheng H, Zhu Y-M, et al. SIMULTANEOUS ACCURATE DETECTION OF PULMONARY NODULES AND FALSE POSITIVE REDUCTION USING 3D CNNS[M]. 2018: 1005-1009.

[7] Collobert R, Weston J, Bottou L, et al. Natural Language Processing (Almost) from Scratch[J]. Journal of Machine Learning Research, 2011, 12: 2493-2537.

[8] Jia F, Lei Y, Lin J, et al. Deep neural networks: A promising tool for fault characteristic mining and intelligent diagnosis of rotating machinery with massive data[J]. Mechanical Systems and Signal Processing, 2016, 72-73: 303-315.

[9] Liang S, Yin S, Liu L, et al. FP-BNN: Binarized neural network on FPGA[J]. Neurocomputing, 2018, 275: 1072-1086.

[10] Cheng Y, Wang D, Zhou P, et al. A Survey of Model Compression and Acceleration for Deep Neural Networks. arXiv e-prints, 2017.

[11] Hubara I, Courbariaux M, Soudry D, et al. Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations[J]. Journal of Machine Learning Research, 2018, 18.

[12] Loshchilov I, Hutter F. Decoupled Weight Decay Regularization. arXiv e-prints, 2017.

[13] Cun Y L, Denker J S, Solla S A. Optimal brain damage[C]. Proceedings of the 2nd International Conference on Neural Information Processing Systems, 1989: 598-605.

[14] Hassibi B, Stork D G. Second Order Derivatives for Network Pruning: Optimal Brain Surgeon[C]. Advances in Neural Information Processing Systems 5, [NIPS Conference], 1993: 164-171.

[15] Srinivas S, Venkatesh Babu R. Data-free parameter pruning for Deep Neural Networks. arXiv e-prints, 2015.

[16] Han S, Pool J, Tran J, et al.: Learning both Weights and Connections for Efficient Neural Networks, Cortes C, Lawrence N D, Lee D D, Sugiyama M, Garnett R, editor, Advances in Neural Information Processing Systems 28, 2015.

[17] Rigamonti R, Sironi A, Lepetit V, et al.: Learning Separable Filters, 2013 Ieee Conference on Computer Vision and Pattern Recognition, 2013: 2754-2761.
[18] Jaderberg M, Vedaldi A, Zisserman A. Speeding up Convolutional Neural Networks with Low Rank Expansions. arXiv e-prints, 2014.
[19] Howard A G, Zhu M, Chen B, et al. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. arXiv e-prints, 2017.
[20] Iandola F N, Han S, Moskewicz M W, et al. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. arXiv e-prints, 2016.
[21] Shang W, Sohn K, Almeida D, et al. Understanding and improving convolutional neural networks via concatenated rectified linear units[C]. Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, 2016: 2217-2225.
[22] Li H, Ouyang W, Wang X. Multi-Bias Non-linear Activation in Deep Neural Networks. arXiv e-prints, 2016.
[23] Hinton G, Vinyals O, Dean J. Distilling the Knowledge in a Neural Network arXiv[J]. arXiv, 2015: 9 pp.-9 pp.
[24] Mishra A, Marr D. Apprentice: Using Knowledge Distillation Techniques To Improve Low-Precision Network Accuracy. arXiv e-prints, 2017.
[25] Romero A, Ballas N, Ebrahimi Kahou S, et al. FitNets: Hints for Thin Deep Nets. arXiv e-prints, 2014.
[26] Han S, Mao H, Dally W J. Deep Compression: Compressing Deep Neural Networks with Pruning, Trained Quantization and Huffman Coding. arXiv e-prints, 2015.
[27] Wenlin C, Wilson J T, Tyree S, et al. Compressing Neural Networks with the Hashing Trick arXiv[J]. arXiv, 2015: 10 pp.-10 pp.
[28] Courbariaux M, Bengio Y, David J-P: BinaryConnect: Training Deep Neural Networks with binary weights during propagations, Cortes C, Lawrence N D, Lee D D, Sugiyama M, Garnett R, editor, Advances in Neural Information Processing Systems 28, 2015.
[29] 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 e-prints, 2016.
[30] Rastegari M, Ordonez V, Redmon J, et al.: XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks, Leibe B, Matas J, Sebe N, Welling M, editor, Computer Vision - Eccv 2016, Pt Iv, 2016: 525-542.
[31] Chenzhuo Z, Song H, Huizi M, et al. Trained Ternary Quantization arXiv[J]. arXiv, 2016: 9 pp.-9 pp.
[32] Fengfu L, Bin L. Ternary Weight Networks arXiv[J]. arXiv, 2016: 9 pp.-9 pp.
[33] Zhou S, Wu Y, Ni Z, et al. DoReFa-Net: Training Low Bitwidth Convolutional Neural Networks with Low Bitwidth Gradients. arXiv e-prints, 2016.
[34] Tang W, Hua G, Wang L. How to Train a Compact Binary Neural Network with High Accuracy?[M]. 2017.
[35] Bethge J, Bornstein M, Loy A, et al. Training Competitive Binary Neural Networks from Scratch. arXiv e-prints, 2018.
[36] Zechun L, Baoyuan W, Wenhan L, et al. Bi-Real Net: Enhancing the Performance of 1-Bit CNNs with Improved Representational Capability and Advanced Training Algorithm[M]. 2018: 747-63.

[37] Deng L, Jiao P, Pei J, et al. GXNOR-Net: Training deep neural networks with ternary weights and activations without full-precision memory under a unified discretization framework[J]. Neural Networks, 2018, 100: 49-58.
[38] Sklyarov V, Skliarova I, Silva J. On-Chip Reconfigurable Hardware Accelerators for Popcount Computations[J]. International Journal of Reconfigurable Computing, 2016.
[39] El-Qawasmeh E. Beating the popcount[M]. 9. 2003.
[40] Simonyan K, Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition[J]. Computer Science, 2014.
[41] Ioffe S, Szegedy C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. arXiv e-prints, 2015.
[42] Pearson K. LIII. On lines and planes of closest fit to systems of points in space[J]. The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science, 1901, 2(11): 559-572.
[43] Van Der Maaten L, Hinton G. Visualizing Data using t-SNE[J]. Journal of Machine Learning Research, 2008, 9: 2579-2605.
[44] Dunne R A, Campbell N A. On the pairing of the softmax activation and cross-entropy penalty functions and the derivation of the softmax activation function[M]. 1997: 181-5.
[45] 牟帅. 基于位量化的深度神经网络加速与压缩研究[D].华南理工大学,2017.
[46] 陈昀,蔡晓东,梁晓曦,王萌.权重量化的深度神经网络模型压缩算法[J].西安电子科技大学学报,2019,46(02):132-138.

致谢

经过这八个月来的努力，毕业设计工作顺利完成，完成的情况超过我自己的预想。在去年的这个时候，自己对于深度学习网络几乎一无所知。去年下半年，自己选修自动化系的机器学习及图像处理相关课程，自学 Pytorch 和 Tensorflow 深度学习框架，再到这个学期不断阅读这个领域论文，对于论文中的实验进行复现，然后到提出自己的创新想法，进行实践。自己在这个未来要从事的领域成长不少。深度学习，包括深度学习网络压缩量化，是一个很广阔的领域，有很多难题和机理性问题没有解决，我所做的量化网络准确率问题只是很小一部分，我希望毕设工作是我整个机器学习研究一个新的起点，未来我还需要不断进行探索。

毕设工作的超预期完成，离不开各位老师、学长学姐的帮助、支持和鼓励。我首先要感谢我毕设指导老师杨杰老师，以及实验室的黄晓霖老师和屠恩美老师。一方面他们为我毕业设计的工作提供 GPU 资源，能够让我有工具完成实验、实践自己的想法；另外一方面教授我机器学习基础知识和概念，在毕设关键时候提出了很多建设性的指导意见，对我提出想法给出了很多有意义的观点和实现方案，鼓励我将毕业设计工作完成得更好。其次我要感谢楚天舒学长，他在我这次毕业设计工作中给了很多帮助，当我在具体程序细节上出现问题时，他和我一起找出问题，解决问题。并且他给我传授了很多深度学习网络训练方面的实践经验，这些经验对我以后的研究工作也将适用。我还要感谢毛义梅老师，她可以说是我本科阶段遇到最好老师之一，毕设阶段她帮助我解决了跨系做毕设手续上的问题，并在很多流程和时间节点上进行提醒。在最后毕业论文阶段，她对于我的论文提出了很多修改意见，使得我论文内容和格式上更加完善。最后还要感谢工程训练中的老师们，论文中有几个实验是利用工程训练中心提供的 GPU 资源完成的，希望老师们继续能够提供这项服务，让更多机器学习研究者受益。

回顾大学四年，自己成长不少，目标更加坚定，脚步更加踏实。在研究生阶段，自己将要去自动化系就读。在这里首先感谢母系仪器系对我四年来的栽培，感谢仪器系的老师。特别感谢蔡萍老师、闫浩老师以及颜国正老师。他们给了我实践的机会，教会我如何阅读文献、如何进行科研、如何进行论文写作，在我决定研究生阶段跨系时给了很多支持、鼓励和建议。无论我在哪里，我都不会忘记仪器系和仪器系老师们曾经对我的厚爱和帮助。其次，感谢徐添翼、马仲柯、张徐玮、郑超等仪器系的学长学姐，他们在我迷茫的时候给了我启迪，在我失落的时候给了我鼓励，在学业上遇到困难时给了我帮助。我还想在论文致谢中重申一点，自己选择跨越行业并不是对于仪器行业或者仪器系的嫌弃和失望，而是觉得自己更加喜欢和适合从事偏软和算法的方向，自己愿意去尝试新的领域。我一直认为，仪器依然是我们国家科研工业中很重要一环，急需要提高。我也期待交大仪器系能够蒸蒸日上，大量中国自产的仪器能够进军世界市场，占据一席之地。

我也要感谢和我一起奋斗的同学和朋友，包括我的三位室友。他们在四年也帮助了我很多，很幸运能够和大家一起学习，讨论问题。虽然大家各奔东西，处于不同行业，但“聚是一团火，散作满天星”，大家一起继续加油，在各行各业上做出自己贡献。

最后，我要感谢我的母亲和外婆。她们是这个世界上最关心我的人，在我遇到困难挫折的时候给了很多鼓励。她们乐观向上人生态度对我也是一种激励。在我本科毕业之际，真的想对她们说，辛苦你们了，真的很爱你们。感谢所有爱我的人，在新的前途上，我将更加勤勉努力，脚踏实地，提升自我，在新的领域做出自己成绩。

Research On Modeling And Accuracy Improvement Of Hybrid-bit Quantized Neural Network

Deep Learning is superior to other state-of-the-art machine learning algorithm by virtue of its powerful fitting ability. It is widely used in many fields, such as image recognition, natural language processing and so on. However, there are several drawbacks for deep learning, among which is its large size and huge demand of computing resources and storage space. Training and deploying deep neural network often requires GPU, but its energy consumption is so large that it is not the best choice to use in some power-restricted area. Large deep neural networks are difficult to deploy directly on devices with limited storage area and weak computation ability, such as mobile phones and some embedded systems like FPGA, DSP. In order to solve the problems of large memory consumption and calculated time consumption, many compression ideas have been put forward, among which a recent network quantization scheme , converting the parameters in the deep network from 32-bit floats to 1,2 or 8 bit number, outperforms other network compression schemes in memory consumption and time consumption. In terms of storage space, it is 32 times smaller than other network. And in terms of operation time, much time consumption could be saved because it converts the multiply-addition operation to XNOR-popcount operation. It establishes the connection between deep learning and hardware, making researchers rethink about neural network from aspects of hardware. A problem for network quantization is that there exists a loss in the prediction accuracy compared to full-precision network. Many attempts to minimize the loss have been conducted, and many advanced quantized neural networks have been proposed, including XNOR-Net, TNN and DoReFa-Net. This passage concentrates on the method of reducing the prediction gap between quantized neural network and full-precision network.

The main research of the thesis are as follows:

(1) According to the quantization methods and training processes of binarized and ternarized network proposed by other reasercher, we construct binarized and ternarized network and train them on CIFAR-10 dataset aimed at obtaining the testing accuracy of 85%.

(2) Reproduce the experimental results of multiple-bit neural network. Make research on the distinguish ability of output characteristics for different layer. Based on the experimental results above, we propose hybrid-bit neural network structure. We train it on CIFAR-10 and CIFAR-100 dataset and compare the performance of it with that of binarized and ternarized neural network.

(3) Design the regularizers for binarized and ternarized neural network. Train the quantized network with and without regularizer on CIFAR-10 dataset. Analyze the impact on performance of quantized network for regularizer using accuracy curve and weight histogram.

(4) Make research on how to connect the knowledge distillation with quantized neural network. From the aspects of accuracy and category, we analyze the impact on performance

of quantized neural network by using different parameters in knowledge distillation and student networks with different learning ability.

Above all, the thesis puts forward the framework of binarized and ternarized neural network, from aspects of quantized method, training methods and bit operation. The quantization methods of binarized and ternarized neural networks are generally divided into two category: without coefficient and with coefficient. The quantization function of binarized network without coefficient is sign function. An important problem for the training of such deep neural network is the backward propagation process. We use the rectangular pulse to mimic the impulse function, the gradient of the quantization function. And the quantization function for ternarized neural network without coefficient is the doubled step function between {-1,0,1}. The mimic for gradient of it is two rectangular pulses. The formulas of bit operation for binarized and ternarized neural network are inferred in the thesis. The XNOR-popcount rather than multiply-addition calculation could accelerate the operation speed for deep learning algorithm. We compare the prediction accuracy of binarized, ternarized and full-precision neural network on CIFAR-10 datasets, using VGG-like net and ResNet, two uniform deep network frameworks. The gap between quantized and full-precision neural network is not large on small dataset, like CIFAR-10. The performance of ternarized neural network is better than binarized neural network, because of the sparse region in ternarized method. Binarized and Ternarized neural network could extract the boundary information in a picture, but some color information loses during quantization, which is important for classification in latter layer. And then we compare the performance of the quantized neural networks with and without coefficients. The results show that the coefficients have little effect on the improvement of quantized neural network because of batch normalization layer. Although the overfiting problem is not as severe as it in full-precision network, the regularization might help improve prediction accuracy. We design different regularizers for binarized and ternarized neural networks on the principle of making the float weight parameters approach several discrete quantized values, to make the quantized network much more sparse under the premise of minimizing the quantization loss. We conduct the experiment on the binarized and ternarized neural network with different regularizers and regularization coefficients. The results show that the histogram of weight parameter distribution satisfy our assumption, which is more weight parameters approach the discrete values. However, the testing accuracy doesn't improve, which means that our assumed weight distribution for quantized neural network is not the most suitable.

And then, the thesis illustrates a new quantized network framework put forward by us, the name of which is hybrid-bit quantized neural network. We assume that different layers have different demand of quantized bits. Based on this assumption, we propose that the hybrid-bit quantized neural network. The idea of our network framework originates from two experiments. The first experiment is to study the testing accuracy varies with the quantization bits using DoReFa-net. The results show that when the quantization bit for gradient is larger than 6, the testing accuracy mainly depends on the quantization bits of weights and activations. Using weights and activations with more bits could improve testing accuracy. This inspires us to use more bits in our network. The other experiment is about the ability to divide the layer output characteristic into different categories. We use PCA and t-sne, two dimension reduction methods, to describe the high dimensional output characteristic. The results show that the output characteristics in latter layer from different categories are easy to distinguish. And this inspires us

to construct the network whose quantization bit decays from front to back layer. These two experiments provide solid theoretical fundamentals for our network framework. And next, we illustrates two characteristic of our proposed network scheme, one is that quantization bit decays from front to back layer and the other is that the average quantization bit in the network is between one to two. The training process is almost the same as that of the quantized network above, and the main difference is the introduction of GrapMap layer, which could provide different learning rate in the parameter update stage for the layers with different quantization bit. We compare the performance of binarized, ternarized and our hybrid-bit quantized neural network on CIFAR-10 and CIFAR-100 using VGG-like net and ResNet-20. The results show that our network outperforms binarized neural network so much, and the performance of it approaches or even surpasses ternarized neural network.

In the end, we attempt to introduce some prior knowledge to the training of quantized neural network and improve the testing accuracy of quantized neural network without introducing any additional bit. We combine the knowledge distillation, a method to train small model, and network quantization. The prior knowledge comes from the knowledge distilled by a teacher model, which is cumbersome but performs better in training and testing datasets. In training process, for the same sample input, the teacher model produces the soft target, which is calculated by dividing the last linear output by the temperature coefficient and then softmaxing. The soft target provides smoother label and contains more information. The student model produces the soft output and the hard output, calculated simply by softmax. And the loss function contains two parts, one is the KL divergence of soft outputs of teacher and student model, the other is the crossentropy between hard output of student network and one-hot vector. In the experiment, we use full-precision ResNet-30(inflate=4) as teacher model, and use three quantized networks with different study ability as student models. We could find that the knowledge distillation method helps improve accuracy by 1% for those two quantized networks with relatively strong learning ability, and the accuracy of the quantized network with weakest learning ability hardly improves. We draw the curve about the rate of improvement for prediction accuracy from different categories. The curve of hybrid-bit ResNet-20(inflate 4) is similar to that of full-precision teacher network, meaning the network has learned the knowledge distilled from teacher, however, that of hybrid-bit ResNet-20(inflate=1) is hardly unlike that of full-precision teacher network for lack of learning ability. Learning ability of quantized network is a key element to determine whether knowledge distillation could help improve testing ability or not.