深 度 学 习 模 型 量 化 技 术 报 告

目 录

[深度学习量化技术报告 1](#_Toc31741)

[1 绪论 2](#_Toc31827)

[1.1 背景介绍 2](#_Toc21615)

[2 量化原理 1](#_Toc15857)

[2.1 量化模型的优势 1](#_Toc29781)

[2.2 使用定点运算 1](#_Toc24348)

[2.3 均匀量化与非均匀量化 2](#_Toc226)

[2.4 对称量化与非对称量化 3](#_Toc4467)

[2.5 总结 4](#_Toc27013)

[参考文献 5](#_Toc16819)

1 绪论

1.1 背景介绍

在计算机视觉、自然语言处理任务上深度学习模型有着大量的需求与应用，如最近很火的自动驾驶领域以及大语言模型等。通常参数量较大的模型由于有更多的参数，理论上可以学习更复杂的特征表示从而在某些任务上获得更好的性能，但这也意味着模型占用的存储空间更多以及计算时间更长。

而在实际应用中如移动端、嵌入式、物联网设备的部署中，往往会受限于硬件设备的算力、内存、功耗等要求。模型需要满足参数量少、计算复杂度低、耗电量低等条件，模型压缩和加速技术因此诞生。

模型压缩技术是一种优化手段，旨在减少深度学习模型的大小、计算复杂度以及内存需求，同时尽量保持模型的预测精度不变或仅有轻微下降，这项技术对于部署模型到资源受限的设备（如移动设备、嵌入式系统）至关重要。模型压缩与加速技术可以根据训练期间结构的优化、后端推理框架优化以及硬件加速方案分为三类[1]：

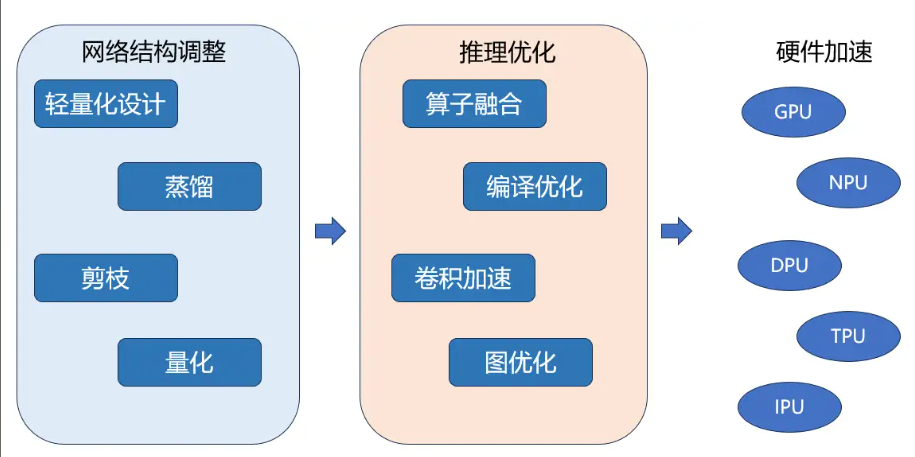


图1-1 量化技术方案路线

Fig. 1-1 Quantitative Technology Solution Route

其中网络结构调整中，各技术的简单介绍如下：

轻量化设计：主要通过精简的网络结构设计来减少计算量，如一些轻量化网络结构设计MobileNet、ShuffleNet等，依靠分组卷积、分解卷积、Bottleneck结构等减少计算复杂度。

模型蒸馏：核心目标是将一个复杂、计算量大的预训练模型（教师模型）的知识迁移到一个更为轻量级、计算效率更高的模型（学生模型）中，同时尽量保持学生模型的预测性能接近教师模型。这一过程不仅能够减少模型的大小和推理时间，还便于将大型模型部署到资源受限的设备上。

模型剪枝：旨在通过移除神经网络中对输出贡献较小的权重或整个结构单元（如神经元、滤波器、通道等），来减少模型的大小和计算复杂度，同时尽可能保持模型的性能。剪枝可以分为两大类：结构化剪枝和非结构化剪枝。

模型量化：模型量化可以减少模型的内存占用和加速推理过程，使之能在资源受限的设备（如移动设备、嵌入式系统）上高效运行。这一过程涉及到将模型中的权重和激活值从高精度的浮点数转换为低精度的表示形式，通常是如int8、int16类型的整形数据类型。量化的核心目的是在不显著牺牲模型准确性的前提下，实现模型的高效部署。

2 量化原理

2.1 量化模型的优势

深度学习模型不管是训练还是推理时，从输入、权重到中间特征的计算数据类型通常为浮点类型如float32，量化（Quantization）技术将参与运算的浮点数转换成了定点数。量化模型之所以计算更快，主要归因于以下几个方面：

硬件加速：针对人工智能应用设计的硬件（如NPU、TPU和其他AI加速器），对整数量化操作进行了优化。相比于浮点运算，整数运算通常有专门的硬件指令集支持，能够并行处理更多数据，提高计算效率。整数运算单元通常比浮点运算单元功耗更低，进一步提升能源效率。

内存带宽和存储效率：量化模型由于使用低精度整数表示，相比浮点模型，所需内存带宽和存储空间大幅减小。这意味着在内存访问和数据传输过程中，量化模型可以更快地加载和处理数据，减少了内存瓶颈，提升了整体计算速度。

计算复杂度简化：量化减少了每个操作的位宽，比如从32位浮点数降到8位整数，计算复杂度相应降低。这不仅减少了计算资源的消耗，还使得单个计算操作更快完成。

能耗降低：由于整数运算的能量消耗低于浮点运算，特别是在大规模部署的设备上，如移动设备和IoT设备，量化模型的低功耗特性尤为重要，有助于延长电池寿命。

2.2 使用定点运算

量化技术的核心就是如何用定点运算去表示浮点运算[2]，由于浮点数的取值数量远远大于定点数，所以转为定点数后还需要想办法让模型的输出结果不出现太多的掉点。以float32和uint8（0~255）之间的数据转换为例，模型中某个浮点数据如某一层的权重）值域范围在之间，要把这个数据映射到0~255上（即量化过程），则换算公式表示如2.1所示：

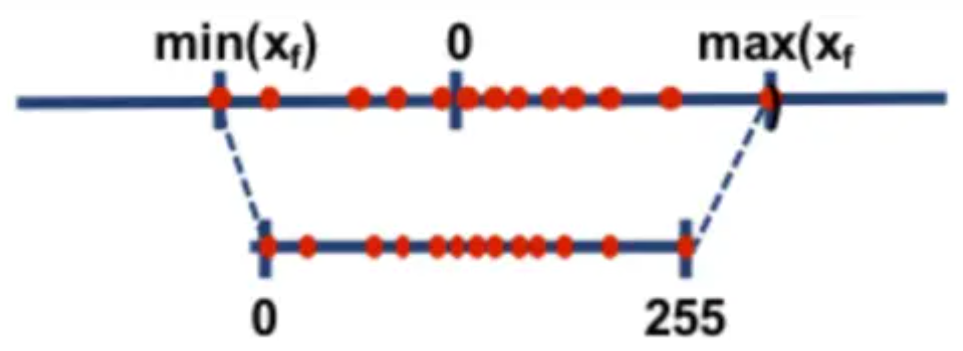


图2-1 定点运算可视化图

Fig. 2-1 Visualization diagram of fixed-point operation

令S（scale）代表float数据的范围，Z（zero point）代表float数据的零点转换到uint数据时对应的位置，如2.2和2.3所示：

最终浮点数量化到定点数的转换公式可以通过S和Z表示，如2.4和2.5所示：

而要想把定点数复原回浮点数，只需要倒推回去就可以了（即反量化过程），如2.6所示：

可以发现在量化过程中由于取整(round)和截断(clamp)操作，反量化后的浮点数据很难完全复原回原始的浮点数据大小。这是由于浮点类型可以表示的数据层次数量多于定点数，在量化过程中数据层次不可逆的降低后很难再破镜重圆，因此量化-反量化的过程是一定会存在精度损失的。

2.3 均匀量化与非均匀量化

在2.2节简单介绍了定点数与浮点数转换的其中一种方式，这种转换方式其实是属于均匀非对称量化，也是工业部署中最常见的一种[3]。接下来补充介绍一下均匀与非均匀量化的概念，它们主要的区别只是在于量化间隔的分配方式：

均匀量化：将连续的实数值信号范围分割成一系列等间隔的小段，每个小段对应一个离散的量化值。这意味着不论信号的大小，每个量化级别的宽度是固定的。这种方法简单易实现，但可能导致小信号区域的量化噪声相对较大，因为固定大小的量化间隔对于幅值较小的信号来说，其相对误差较大而且降低了信噪比。

非均匀量化：非均匀量化则是根据信号的统计特性或分布，动态调整量化间隔的大小。通常在信号幅值较小时采用较小的量化间隔，而在信号幅值较大时采用较大的量化间隔。这种方法通过在信号幅值较低、变化更为敏感的区域使用更精细的量化，提高了小信号的量化精度和信噪比，而对大信号的量化误差影响较小。

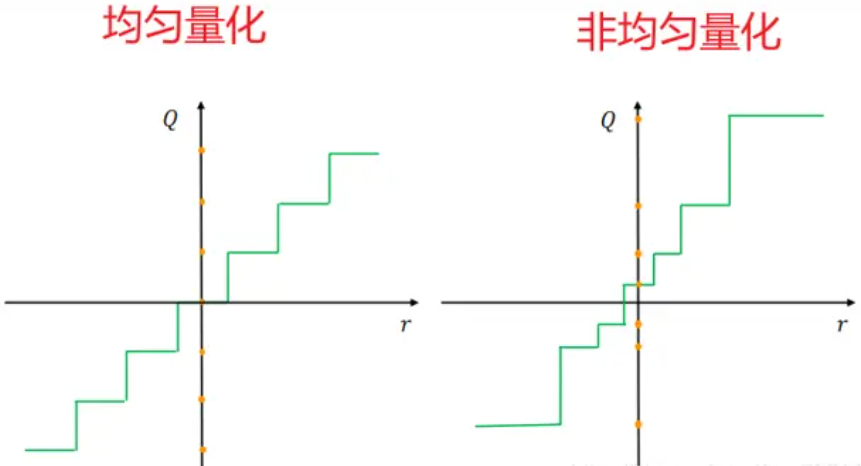


图2-1 两种量化的信号图

Fig. 1-1 Two types of quantized signal graphs

均匀量化和非均匀量化都是为了在保证模型性能的同时，减少模型的存储和计算需求。均匀量化操作简便，但可能在信号动态范围广泛的应用中牺牲精度；而非均匀量化虽然复杂度稍高，但能更有效地利用量化级别，尤其是在处理信号幅值变化范围大的情况下，能显著提升量化效率和模型性能。

2.4 对称量化与非对称量化

对称量化假设量化范围是对称的，即定点数的量化级别围绕0点均匀分布[4]。这种量化方式下，正负两边的量化间隔相同，且量化值包括一个零点和相同数量的正负数值，比如float32到int8（-127~128）的转换方式就是对称量化，此时零点Z也就等于0了。对称量化通常用于处理权重的量化，因为权重往往分布在零附近且对称性较好。

非对称量化则允许量化范围不对称，通常用于激活值量化，因为它可以更灵活地适应激活值的实际分布。在这种量化方式中，量化区间并不强制包含零点，而是可以根据数据的实际分布来调整，因此正负范围可以不同。

接2.2节的非对称均匀量化推导，以float32与int8之间的转换为例，对称均匀量化的公式如2.7和2.8所示：

由于对称量化中浮点数和定点数的零点都在数据范围的中心点，因此zero point就直接等于零，最终的量化表达式如2.9和2.10所示：

反量化表达式如2.11所示：

2.5 总结

量化技术，作为一种广泛应用于多个领域的技术手段，其重要性日益凸显。在深度学习领域，量化技术作为模型压缩的一种常用方法，通过降低模型参数与运算的精度（如从FP32降低至INT8），不仅显著提升了模型在CPU/GPU等硬件上的推理计算效率，减少了计算成本，还有效减小了模型的体积，便于在边缘设备上部署。这一过程虽然是有损的，但通过量化感知训练（QAT）等策略，可以最大限度地减少量化后模型的精度损失[5]。

在投资领域，量化技术同样发挥着重要作用。它涵盖了投资品种选择、投资时机选择、股指期货套利、商品期货套利、统计套利和算法交易等多个方面。通过运用先进的数学模型、统计方法和计算机技术，量化投资能够处理海量的市场信息，快速捕捉市场机会，制定并执行投资策略，为投资者带来稳定的回报。

然而，量化技术的发展也面临着诸多挑战。首先，数据质量是量化交易的核心，如何获取和处理高质量的数据是量化交易面临的重要挑战之一。其次，随着量化交易策略的日益复杂，技术难题也随之增加，如何提高算法的效率和准确性成为了一个亟待解决的问题。此外，量化交易还受到法规限制的影响，各国政府可能会出台更多法规来规范量化交易行为。最后，随着量化交易的普及，市场竞争也日益激烈，如何在激烈的竞争中保持优势是量化交易者需要面对的重要课题。

总的来说，量化技术以其独特的优势在多个领域得到了广泛应用，并展现出强大的生命力和广阔的发展前景。然而，其发展过程中也面临着诸多挑战和问题需要解决。未来，随着技术的不断进步和市场的不断规范，相信量化技术将会更加成熟和完善，为更多领域的发展贡献力量。

参考文献

1. Dong Y, Ni R, Li J, et al. Learning accurate low-bit deep neural networks with stochastic quantization[J]. arxiv preprint arxiv:1708.01001, 2017.
2. Jacob B, Kligys S, Chen B, et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2018: 2704-2713.
3. Krishnamoorthi R. Quantizing deep convolutional networks for efficient inference: A whitepaper[J]. arxiv preprint arxiv:1806.08342, 2018.
4. Gong R, Liu X, Jiang S, et al. Differentiable soft quantization: Bridging full-precision and low-bit neural networks[C]//Proceedings of the IEEE/CVF international conference on computer vision. 2019: 4852-4861.
5. Esser S K, McKinstry J L, Bablani D, et al. Learned step size quantization[J]. arxiv preprint arxiv:1902.08153, 2019.