联邦学习环境下的高效通信方法研究

目录

[联邦学习环境下的高效通信方法研究 1](#_Toc102856584)

[一、 摘要 1](#_Toc102856585)

[二、 引言 1](#_Toc102856586)

[1. 研究背景 1](#_Toc102856587)

[2. 国内外研究现状 2](#_Toc102856588)

[2.1基于降低模型更新频率的优化方法 2](#_Toc102856589)

[2.2基于模型压缩的优化方法 3](#_Toc102856590)

[2.3基于客户端选择的优化方法 3](#_Toc102856591)

[3. 研究内容 3](#_Toc102856592)

[4. 论文组织结构 4](#_Toc102856593)

[三、 相关工作 4](#_Toc102856594)

[1. 剪枝 4](#_Toc102856595)

[3.1以权重为度量对象的剪枝方法 5](#_Toc102856596)

[3.2以激活单元为度量对象的剪枝方法 5](#_Toc102856597)

[3.3以梯度为度量对象的剪枝方法 6](#_Toc102856598)

[2. 量化 6](#_Toc102856599)

[四、 算法的分析与设计 7](#_Toc102856600)

[1. 算法系统背景描述 7](#_Toc102856601)

[2. 算法定义 7](#_Toc102856602)

[2.1剪枝算法 7](#_Toc102856603)

[2.2量化算法 10](#_Toc102856604)

[3. 算法概要设计 14](#_Toc102856605)

[3.1 联邦学习环境中客户端算法的概要设计 14](#_Toc102856606)

[3.2 联邦学习环境中服务器端算法的概要设计 15](#_Toc102856607)

[3.3 剪枝算法的概要设计 15](#_Toc102856608)

[3.4 量化算法的概要设计 15](#_Toc102856609)

[4. 算法详细设计 16](#_Toc102856610)

[4.1 联邦学习环境中客户端算法的详细设计 16](#_Toc102856611)

[4.2 联邦学习环境中服务器端算法的详细设计 16](#_Toc102856612)

[4.3 剪枝算法的详细设计 17](#_Toc102856613)

[4.4 量化算法的详细设计 17](#_Toc102856614)

[六、结论 17](#_Toc102856615)

[七、参考文献 18](#_Toc102856616)

# 摘要

联邦学习允许将用户数据保存在用户设备上，并借助用户设备固有的计算能力来训练一个共享的机器学习模型。然而，在联邦学习环境中，用户设备和参数服务器反复交换大量的模型参数，通信次数高、通信数据的总量大，造成了较大的通信开销，也成为了联邦学习的主要瓶颈之一。如何在保证收敛速率、确保模型精度的前提下降低客户端与服务器的通信开销，是联邦学习研究的研究热点。

本文对联邦学习环境进行了模拟，FATE框架搭建了联邦学习环境，并且参考Distiller项目将几种经典的剪枝和量化方法集成到FATE框架中，在联邦学习的情境下使用不同的压缩策略对多个神经网络模型（AlexNet、ResNet等）进行了模型压缩，并根据实验结果对压缩效果以及联邦学习聚合效率进行了分析。结果显示，通信效率在模型精度损失很小、收敛速率降低幅度很小的情况下得到了较大提升。

# 引言

## 研究背景

在物联网时代，智能手机、平板以及物联网设备无时无刻不产生着大量数据。使用这些数据来训练机器学习模型可以很好地提升用户体验。然而，由于用户数据的隐私性，将这些数据上传到中心服务器再使用传统的机器学习算法进行集中训练是不可行的。出于保护用户隐私的考虑，联邦学习范式应运而生。它允许将用户数据保存在用户设备上，并借助用户设备固有的计算能力来训练一个共享的机器学习模型。

联邦学习的一般过程如下：首先参数服务器将当前模型发送给各个用户设备，用户设备使用本地数据来训练出一个新模型，然后将模型更新发送到参数服务器，参数服务器将这些模型更新聚合后得到最新的模型，重复以上步骤直到模型收敛。

从以上流程可以看出，用户设备和参数服务器反复交换大量的模型参数，通信次数高、通信数据的总量大，造成了较大的通信开销；再者，由于绝大多数用户设备是通信带宽有限、电量有限的移动设备，使得联邦学习中的通信是比较慢速且不稳定的，这些因素导致了通信开销是联邦学习的主要瓶颈。因此，在联邦学习环境中，如何降低通信开销、提高通信效率，是近来联邦学习优化算法研究的热点方向。

在联邦学习的实际应用中，对企业来说，联邦学习主要用来解决企业信息化建设中的数据孤岛问题。因此，找到一种高效通信的联邦学习方法，降低通信开销，减轻网络负担，能切实降低企业运营成本，提高企业的经济效益；对用户来说，降低用户设备与参数服务器的通信开销，使得共享模型的训练对用户透明，降低用户设备的关键资源消耗，能更好地提升用户体验。

## 国内外研究现状

关于从通信成本角度优化的联邦学习算法，国内外已有以下几种优化角度：

### 2.1基于降低模型更新频率的优化方法

降低联邦学习环境中的模型更新频率，可以考虑以客户端的计算成本来换取客户端与参数服务器之间的通信成本。

Konečný4通过增加每一轮迭代在每个客户端本地的模型计算次数。与每一轮服务器参数更新只需一次客户端本地计算的FedSGD算法相比，研究结果显示，当数据为非独立同分布时，该算法对于通信成本地减少效果并不明显。因此，实际应用价值不大，需要进一步探究针对非独立同分布数据的优化算法。Li等人5提出了一种更通用的FedProx算法。该算法并不要求每一次迭代时不同客户端的本地计算次数统一，因此，FedProx算法可以动态更新不同客户端每一轮需要本地计算的次数，充分利用客户端的计算资源，使得算法更适合客户端数据为非独立同分布的场景。

### 2.2基于模型压缩的优化方法

部分优化算法以减少每一轮通信的参数量，如通过模型压缩的技术（量化、下采样等）降低通信数据量来提高通信效率。

Jakub Konečný等人6提出了结构化更新与轮廓更新两种方式来更新服务器参数。结构化更新是指通过提前定义上传模型参数的矩阵结构上传模型，而轮廓更新是指每次更新的参数需要在客户端本地进行压缩编码，从而实现模型的压缩。Caldas等人7考虑的是从服务器到客户端的模型参数传递优化，通过有损压缩的方式来减少从服务器到客户端需要传递的参数数量。当迭代次数保持不变时，模型压缩方法会在一定程度上降低模型准确率。

### 2.3基于客户端选择的优化方法

在联邦学习中，客户端的数量可能非常大，但由于模型分发和重新上传的带宽相当有限，一般只选取部分客户端参与训练过程，因此，客户端选择策略对联邦学习过程中的通信效率至关重要。

Nishio等8提出了数据IID场景下的FedCS算法，其根据累计有效参与值选择模型迭代效率最高的客户端继续聚合更新，以此提高算法的收敛效率，进而降低通信开销。Yoshida等9在启发式算法的基础上提出以一种Hybrid-FL的协议，该协议可处理non-IID的客户端数据，但其一定程度上增加了通信损耗，因为服务器需要通过额外的资源请求选择部分客户端。

## 研究内容

深度神经网络DNN的应用十分广泛。然而为了更强的特征表示能力，DNN也变得越来越大（通过实验数据说明）。越来越多的参数需要很强的存储能力、计算能力、电力等等，这些限制对在移动端的部署构成了巨大挑战。在资源受限的联邦学习环境中，具有超大量参数的模型的聚合和分发加大了服务器端和客户端通信的压力。

尽管DNN模型需要大量参数保证其表征能力，然而近来有研究指出神经网络中存在大量的冗余。这使得压缩神经网络模型（几乎不带来精度的损失）变得可能。

只传递压缩后的模型，能够极大地减少联邦学习环境中机器的通信压力（需要再进行扩充）。因此，在客户端训练好模型后，对训练好的模型进行压缩，将压缩后的模型发送到服务器端执行聚合。

在联邦学习的实际应用中，数据持有方的数据质量和分布是不可控的，无法要求数据持有方的数据满足独立同分布。因此，联邦学习算法需要在非独立同分布数据也表现良好。

## 论文组织结构

本论文共分为5章，每章的组织结构安排和内容如下：

第1章是引言。本章简要介绍了联邦学习的基本知识、减少通信开销的问题以及解决该问题的意义、国内外对于减小联邦学习通信开销的研究现状、本文的主要贡献等。

第2章是相关工作。本章介绍了联邦学习环境下模型压缩的相关知识并提出了其中的问题。

第3章是模型压缩的算法分析与设计。本章在对以前的研究工作进行全面分析的基础上，提出了剪枝和量化算法。

第4章是联邦学习的算法实现。我们利用模拟实验验证了该算法具有较低的时间延迟和较少的通信开销。

第5章结论，给出了本文的结论以及未来工作。

# 相关工作

## 剪枝

剪枝是模型压缩研究的一个子领域。该类方法通常给出一个关于度量对象重要性的评价指标，将不重要的权重剪枝掉。神经网络模型中的度量对象一般有三种：权重、激活单元以及梯度。

### 3.1以权重为度量对象的剪枝方法

《Dynamic Network Surgery for Efficient DNNs》这篇文献中，作者提出了动态网络手术的新型网络压缩方法。它通过在运行时对连接进行剪枝来极大地减少网络复杂度。除此之外，作者向剪枝过程中加入连接拼接操作来避免可能不正确的剪枝，使得剪枝精度可以得到恢复。

《Filter Pruning via Geometric Median for Deep Convolutional Neural Networks Acceleration》这篇文献中，作者关注对卷积层的滤波器进行剪枝。作者经过实验分析，认为基于范数的剪枝标准的有效性依赖于两个需求：滤波器的范数偏差应较大，滤波器的最小范数应很小。前者使得剪枝阈值T的搜索空间足够大，便于寻找合适的剪枝阈值；后者使得剪去的滤波器对推理结果的贡献极小。然而，现实中这两个需求并不能经常满足，因此，作者根据最接近几何平均的滤波器可以由其余滤波器表示这一依据，提出了一种使用几何平均对滤波器进行剪枝的方法FPGM，选择相对贡献最少的滤波器进行剪枝。

《To prune, or not to prune: exploring the efficacy of pruning for model compression》这篇文献中，作者提出了一种适用于多种模型的自动化逐步剪枝技术。作者将大的剪枝的模型和小的稠密的模型进行比较，实验结果显示在占用存储空间相同的前提下，前者的精确度更高。除此之外，作者还提出了一种自动化逐步剪枝的方法，其参考学习率衰减的策略，确定迭代轮次为t时的稀疏性。

### 3.2以激活单元为度量对象的剪枝方法

《An Entropy-based Pruning Method for CNN Compression》这篇文献中，作者通过一种新颖的滤波器剪枝策略对现有的CNN模型同时进行加速和压缩。滤波器的重要性由基于熵的方法进行评估，该方法可以减少中间激活单元的数量。同时，作者还提出了一种新的学习调度策略。无论对哪一层进行剪枝，都对网络进行一个或两个epoch的微调，使得性能得到一定的恢复，并且仅当最后一层被剪枝后，对网络进行多个epoch的重新训练。这样极大减少了重新训练时间，并且还防止了剪枝后的网络在早期阶段陷入局部最优。

《Network trimming A data-driven neuron pruning approach》这篇文献中，作者根据神经元在大型数据集上计算的APoZ（Average Percentage of zeros）指标对神经元进行剪枝，并且使用剪枝前权重的值作为初始化对网络进行重新训练，从而达到模型压缩和设计高效模型的目标。

### 3.3以梯度为度量对象的剪枝方法

《Optimal Brain Damage》和《Second order derivatives for network pruning: Optimal Brain Surgeon》从损失函数出发寻找对损失影响最小的神经元进行剪枝。具体地，将目标函数进行泰勒展开，根据每个神经元剪枝后损失函数的变化量快速计算出神经元的重要程度，从而指导剪枝。

《Faster gaze prediction with dense networks and Fisher pruning》

《Importance Estimation for Neural Network Pruning》

## 量化

《Compressing deep convolutional networks using vector quantization》该文献对向量量化方法，包括二值化方法、基于k均值聚类的标量量化方法、乘积量化方法以及残差量化方法进行了介绍，在图像检索任务中压缩CNN模型，展示了向量量化方法相较于矩阵分解方法的优势。

《Compressing neural networks with the hashing trick》该文献提出了一种新型的网络架构——HashedNets，其借鉴随机权重共享的方法，利用低开销的哈希函数随机均匀地将权重映射到哈希桶中进行共享，从而实现对模型的压缩。

《Deep compression Compressing deep neural networks with pruning,trained quantization and huffman coding》该文献中作者提出了一种深度压缩方式，其包含剪枝、量化以及哈夫曼编码三个阶段。剪枝阶段只学习重要的连接，减少连接的数量；量化阶段进行权重共享，减少每个连接的比特数；哈夫曼编码利用有效权重分布的有偏性进一步压缩模型。

# 算法的分析与设计

## 算法系统背景描述

P1联邦学习环境的介绍，说明其功能架构

P2横向联邦学习环境与纵向联邦学习环境的介绍，引出本文关注点是横向联邦学习。

P3说明模型压缩算法对于提高联邦学习通信效率的必要性。

P4模型压缩算法需要满足的一些要求：模型压缩算法应该在模型精度损失很小以及不减慢收敛速度的情况下大幅度降低模型的大小。

## 算法定义

### 2.1剪枝算法

神经网络有大量的研究表明，神经网络模型是过度参数化的，即包含许多冗余的逻辑以及特征。通过神经网络参数中诱导稀疏性可以一定程度上消除冗余。剪枝就是一种在权重和激活单元中诱导稀疏性的常见的方法。

在模型剪枝中，度量剪枝程度的一个指标是稀疏水平。稀疏水平是一个张量中零元素所占的比例。一个张量x的L0范数，定义为张量x中零元素的个数。



相对于稀疏水平的一个指标是密度，是指一个张量中非零元素（Non-Zero Number，NNZ）所占的比例。显然，对于一个张量来说，Density = 1 – Sparsity。

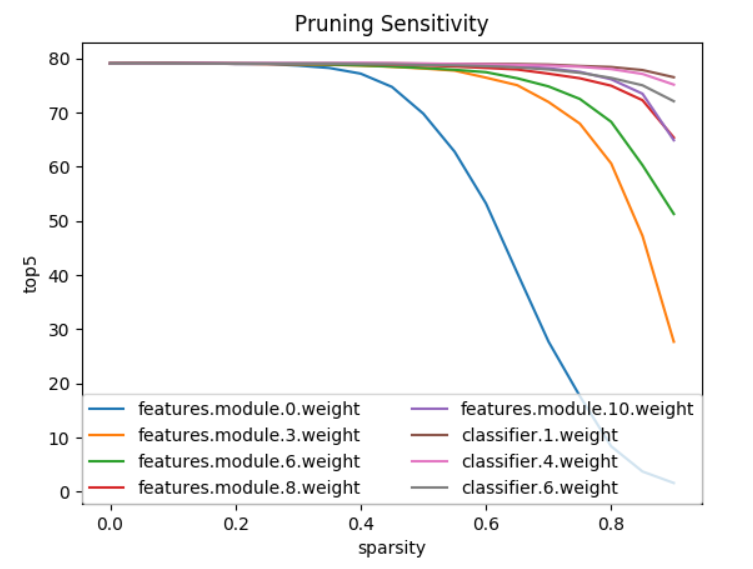
剪枝的对象通常包含权重、偏置以及激活单元。偏置通常是很小的，并且对一个网络层输出的贡献是相对较大的，因此一般不对偏置进行剪枝。激活单元在经过ReLU层的作用后，也通常具有良好的稀疏性，因为ReLU操作会直接将负激活单元的值置为0。而权重的稀疏水平并不高，尽管权重一般都很小，但是通常都是非零值。因此，模型剪枝主要关注的是对模型权重的剪枝。

权重剪枝就是使用一系列方法来增加神经网络中权重的稀疏水平。剪枝需要一个评价指标来选择剪枝掉哪些元素，满足剪枝标准的权重被置0。最常见的剪枝标准是每个元素的绝对值。给定一个阈值δ，如果某个权重的绝对值低于阈值δ，则对该权重进行剪枝。使用L1范数作为剪枝标准的理由是：如果一个权重具有较小的L1范数值，则该权重对最终结果的贡献比较小，因此，可以认为该权重是不太重要，可以被移除。

神经网络模型的训练中，经常使用的一种优化算法是随机梯度下降法SGD。当使用SGD作为优化算法时，对于全连接层而言，如果权重值为0，则其对应的梯度也为0，表明该权重在剪枝后的训练过程中永远不会得到更新。然而对于卷积层来说，即使权重值为0，反向传播阶段计算的对应的梯度也可能不为0。

剪枝方法根据剪枝的粒度可以分为非结构化剪枝以及结构化剪枝。非结构化剪枝是对单个的权重元素进行剪枝，它也被称为是细粒度的剪枝。结构化剪枝是直接对一组元素进行剪枝，以卷积层为例，结构化剪枝是以滤波器为单元进行剪枝。

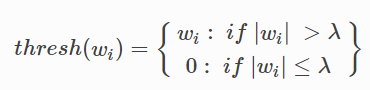
通过剪枝诱导稀疏性的难点在于剪枝阈值的确定。敏感度分析可以根据每一层参数张量的剪枝敏感程度对其进行排序，从而指导各个层剪枝阈值的确定。敏感度分析的基本操作是为一个特定的层设立稀疏水平、执行剪枝，然后在测试数据集上进行评估，并且记录剪枝模型的精度。对所有参数层设定多个稀疏水平，并且执行以上的操作，这样可以得到每个参数层执行剪枝后的模型精度随着稀疏水平的变化情况，从而对剪枝阈值在每个参数层上的确定起到一定的指导作用。



图表 1 对AlexNet执行敏感度分析

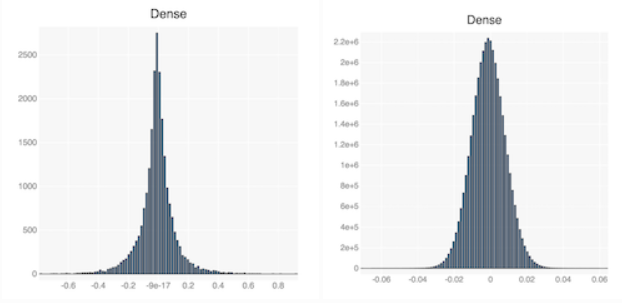
Distiller项目中已经实现了多种剪枝方法：非结构化剪枝包括基于绝对值的剪枝方法、基于敏感性的剪枝方法、基于稀疏水平的剪枝方法以及自动化剪枝方法；结构化剪枝包括：基于L1范数的剪枝方法、基于APoZ的剪枝方法等。

**基于绝对值的剪枝方法。**该方法对每个权重执行一个thresh(·)函数：

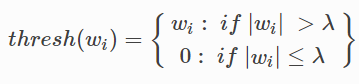


**基于敏感性的剪枝方法。**由于每个层的权重值都不相同，因此为每个层确定一个阈值是十分困难的，然而，可以利用卷积层和全连接层的权重都呈现出均值为0的高斯分布的事实，根据每一层权重的分布特点分别确定该层的剪枝阈值。

下面的图显示了TorchVison预训练的Alexnet模型中第一个卷积层和第一个全连接层权重向量的元素分布，可以看出其分布近似于均值为0的高斯分布。



可以使用权重向量的标准差σ作为是不同权重向量的正则因子。例如，如果一个向量是正态分布，则大约68%的元素的绝对值小于权重向量的标准差。因此，可以将剪枝阈值设置为s\*σ，意为剪枝掉s\*68%的元素。



其中λ=s\*σ，其中σ是第l层权重向量的标准差。

[Learning both Weights and Connections for Efficient Neural Networks](https://arxiv.org/abs/1506.02626)中，作者使用敏感度分析的结果指导s值的确定。一个层越敏感，则该层的s值就越小。因此，敏感度分析是一种经验性的方法，其结果给出了s的一组初始值，更优的值还需要通过实验确定。该方法是不稳定的，因为其稀疏水平依赖于权重分布的收敛模式。

**基于稀疏水平的剪枝方法。**该类方法通过指定目标稀疏水平进行剪枝，而不是指定一个剪枝阈值。在指定目标稀疏水平（如50%）后，该方法对权重进行基于绝对值的升序排序，然后将前50%的权重置为0。该方法比**基于敏感性的剪枝方法**要稳定地多，因为目标稀疏水平并不与元素的绝对值耦合。

### 2.2量化算法

量化指的是减少表示一个数字的比特数目的过程。在深度学习环境中，用于研究和部署的主要的数据格式是32位浮点数，即FP32。然而，由于减少带宽的需要，近来的研究更多地使用低精度的数字格式。有研究表明模型权重和激活单元可以使用8比特的整型表示，同时保持较高的模型精度。

整型与FP32的比较。一个数据格式主要有两种属性：表示范围和可表示的值的数量。对于所有的整型而言，表示范围是[-2^(n-1),2^n-1]，其中n是比特数目。因此，对于INT8来说，表示范围是[-128,127]，可表示值的数目是2^n。而对于FP32来说，表示范围是[-3.4×10^38, 3.4×10^38]，可表示值的数目大约是4.2×10^9。

在深度学习模型中，不同层之间的权重和激活单元的分布通常有着显著差异。为了使用整型格式表示这些不同的分布，一般使用放缩因子将张量的范围映射到整型区间，为了降低映射过程中精度的损失，放缩因子通常为浮点数。

关于量化后的计算，需要处理溢出的问题。在深度神经网络模型中，卷积层和全连接层通常将中间结果存储在累加器中。如果我们对权重、激活单元以及累加器都使用相同的比特带宽，由于整型数据格式能够表示的范围有限，中间结果会很快地溢出。因此，累加器通常使用更高的比特带宽实现。

两个n比特正数相乘的数至多是一个2n比特的数。而在卷积层中，类似的乘法操作还需要累加c\*k^2次，其中c是输入通道的数目，k是卷积核的宽。因此，为了避免溢出，累加器的比特带宽至少应是2n+M，其中M>=log2(ck^2)。

使用INT8量化模型。为了实现FP32格式到INT8格式的模型量化，需要对每个层每个向量计算放缩因子。最简单的情况是将浮点张量的最小值和最大值分别分别映射到整型范围的最小值和最大值。对于权重和偏置来说，这种映射是很容易的，因为当训练结束后，权重和偏置的值都已经确定。然而对于激活单元来说，边界值的确定是比较困难的，因为激活单元的值与输入的值有关。

对于激活单元来说，有离线和在线两种方式确定其边界值。离线方式是在部署模型前收集激活单元的统计数据。当部署模型时，根据训练阶段收集到的激活单元的统计数据来计算激活单元的放缩因子。然而，在实际的推理阶段，激活单元的值可能落在观测到的范围之外，在这种情况下，需要对范围外的值进行钳制，而这可能会导致模型精度的下降。在线方式是在推理阶段动态地计算激活单元的最大值和最小值，这种方式在钳制不会发生，然而，额外的计算开销可能是巨大的。

关于放缩因子的作用范围：最常见的方式是每个层使用一个放缩因子。而当每个通道的权重分布差异较大时，每个通道使用一个放缩因子可能是更有效的。

将一个使用FP32格式训练好的模型直接量化为INT8，而不进行重新训练，这种方法称为训练后的量化。在许多情况下，这种方法会造成较大的精度损失，例如MobieNet等小的模型表征能力较弱，在量化后精度损失较大。

**量化算法的数学定义。**

本论文中使用的量化方法是基于范围的线性量化。线性指的是一个浮点值通过乘以一个常量（放缩因子）进行量化。基于范围的含义是为了计算放缩因子，需要对张量的真实范围进行观察。在最朴素的实现中，使用张量真实的最小值和最大值。另一种方法是使用一些基于张量范围或分布的推导来得到一个更窄的最小值和最大值的范围，来移除一些可能的离群值，这种方法称为基于修剪的量化方法。

**非对称量化模式**

图表, 折线图

描述已自动生成

在非对称模式中，将浮点范围的最小值和最大值分别映射到整型范围的最小值和最大值。除了放缩因子之外，该模式还引入零点（也称为量化偏差、偏移量）来实现这种映射。

使用xf表示原来的浮点张量，xq表示量化后的张量，qx表示放缩因子，zpx表示零点，n表示量化使用的比特数目，则有下式成立：

Word

低可信度描述已自动生成

实际上，通常将使用round(minxfqx)作为zpx的值，这意味着零是由量化范围内的一个整数精确表示，这对有补零操作的层来说是重要的。通过舍入零点，我们有效地“微调”了浮点范围内的最小/最大值，以便获得精确的零量化。

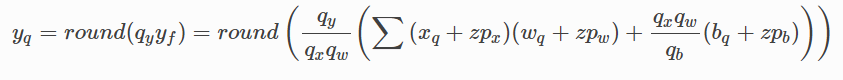
上式使用无符号整数来表示量化范围，即xq∈[0,2^n-1]。也可以使用有符号整数表示，其xq∈[-2^(n-1),2^(n-1)-1]。

使用非对称模式来量化卷积层和全连接层：将输入、输出、权重和偏置分别记为x、y、w和b，卷积层和全连接层的计算有以下的量化表示：

图形用户界面, 应用程序, 表格

描述已自动生成

因此，有：



需要注意：在累加求和的过程中，偏置必须进行重新的放缩来匹配累加器的放缩范围。

**对称量化模式**

图表, 折线图

描述已自动生成

对称量化模式中，不将浮点范围的最小值和最大值映射到量化范围，而选择浮点张量的最大值和最小值中绝对值的最大值作为浮点边界。除此之外，在对称量化模式中，没有必要使用零点。因此，需要量化的浮点范围以及量化后的整型范围都是关于零点对称的。

使用与非对称模式中相同的标记，有下式成立：



卷积层和全连接层在对称模式量化下的表示为：

日程表

低可信度描述已自动生成

因此，有：

图片包含 图示

描述已自动生成

关于两种模式的比较：两种模式的主要差别是简易性和量化范围利用率的折中。

* 当使用非对称量化模式时，量化范围是完全利用的。因为我们精准地将浮点范围地最小值和最大值映射到了量化范围的最小值和最大值。而使用对称量化模式时，如果浮点范围的分布偏向某一侧，可能会导致量化范围中比较多的区域被未观测到的浮点范围的值所占据。比较极端的例子是ReLU层：当经过ReLU层后，张量是非负的，使用对称模式量化，会损失浪费掉一个比特。
* 另一方面，在卷积层和全连接层中，对称量化模式的实现是相当简单的。在非对称模式中，需要执行和零点相关的一些计算，可能会带来额外的计算负担。

其他的一些特性：

* 移除离群值（仅支持训练后的量化）。在某些情况下，激活单元的浮点范围内包含离群值。使用包含离群值的范围进行量化会降低模型的精度。因此，可以使用以下方法来对激活单元进行修剪
  + 平均：使用批次中的每个样本的最小值和最大值的平均来替换全局的最小值和最大值。
  + 均值+/-N\*标准差：取张量平均值的N个标准差，并且在全局的最小值和最大值范围内。N可进行配置。

## 算法概要设计

### 联邦学习环境中客户端算法的概要设计

在横向联邦学习环境中，客户端的主要任务是在本地数据集上训练模型，并且在聚合轮次将训练好的模型以及该模型在验证数据集上的精度和损失数据提交给服务器端。在服务器端完成模型聚合后，接受服务器端分发的聚合模型，并继续在本地数据集上进行下一轮次的训练。

### 联邦学习环境中服务器端算法的概要设计

横向联邦学习环境中，服务器端的主要任务是接收客户端发送的模型，然后使用模型聚合算法对客户端发送的模型进行聚合，并将聚合后的模型分发给各个客户端。

除此之外，服务器端还需要接收客户端发送的损失数据，根据损失数据判断聚合模型的收敛情况，当模型已经收敛时停止训练，防止过拟合。

### 剪枝算法的概要设计

为了减小模型聚合过程中客户端与服务器端的通信开销，引入模型剪枝算法来提高模型的稀疏性，降低模型的存储占用。

剪枝算法主要作用在模型的训练过程中。当开始一轮epoch训练时，模型压缩调度器会根据配置好的剪枝策略生成权重的掩膜矩阵；当开始每个批次时，模型压缩调度器会对权重矩阵与掩膜矩阵计算哈达玛积，即将需要剪枝的权重置为0。然后在剪枝后的模型上计算输出，并进行模型的更新。

### 量化算法的概要设计

本文使用的量化算法属于线性量化算法，并且仅作用在聚合阶段，即训练后的量化。在聚合阶段，压缩调度器根据训练阶段收集到的模型中各个层的输入和输出的统计数据如最小值、最大值等，将输入、输出和权重从32位浮点数类型量化到8位整型类型，然后向服务器端发送量化后的模型以及权重的量化参数。服务器端根据接收到的客户端的量化模型以及对应的量化参数对各模型执行反量化的操作，再进行模型的聚合。最后，服务器端对聚合后的模型执行量化操作，并将量化后的模型与量化参数分发给各个客户端。

## 算法详细设计

### 联邦学习环境中客户端算法的详细设计

伪代码

BEGIN:

While(not Terminate-Condation): # 当不满足终止条件时，循环

If has\_pruning: 　# 如果有剪枝策略

Prune() 　# 对模型进行剪枝

Train() 　# 训练剪枝后的模型

Valid() 　# 对训练好的模型进行验证

If should\_aggregate:　　　　　# 如果当前轮次需要聚合

SendModel() # 给服务器端发送模型

RecvModel() # 接收服务器端聚合的模型

END # 结束算法

### 联邦学习环境中服务器端算法的详细设计

伪代码

BEGIN:

While(not Terminate-Condation): # 当不满足终止条件时，循环

RecvModels() # 接收客户端传来的模型

AggregateModels() # 对客户端传来的模型进行聚合

SendModel() # 分发聚合后的模型

CheckConvergence() # 进行收敛性的验证

If converged:　　　　　 # 如果模型已经收敛

break # 退出循环

END # 结束算法

### 剪枝算法的详细设计

BEGIN:

sensity = GetSensity() # 获取配置好的敏感度

tensor = GetTensor() # 获取需要剪枝的张量

threshold = tensor.stddev \* sensity # 根据标准差和敏感度计算阈值

mask = CreateMask(threshold) # 根据剪枝阈值创建掩膜矩阵

ApplyMask(tensor,mask) # 执行掩膜操作，即按元素相乘

END # 结束算法

### 量化算法的详细设计

主要函数的伪代码：

BEGIN:

config = GetQuantConfig() # 获取量化的配置

model = GetModel() # 获取需要量化的模型

ReplaceParamLayer(model,config) # 替换模型中的参数层

QuantizeForward(model) # 使用量化模型计算输出

END # 结束算法

ReplaceParamLayer函数的伪代码：

BEGIN

CalculateScales() #　计算量化的相关参数

QuantWeights() #　对权重进行量化

RegisterSacles() #　将量化参数注册到模块的缓存中

END

# 六、结论

本文对联邦学习环境进行了模拟，FATE框架搭建了联邦学习环境，并且参考Distiller项目将几种经典的剪枝和量化方法集成到FATE框架中，在联邦学习的情境下使用不同的压缩策略对多个神经网络模型（AlexNet、ResNet等）进行了模型压缩，并根据实验结果对压缩效果以及联邦学习聚合效率进行了分析。结果显示，通信效率在模型精度损失很小、收敛速率降低幅度很小的情况下得到了较大提升。

# 七、参考文献

1. KONEČNÝ J. Stochastic, distributed and federated optimization for machine learning[J]. arXiv preprint, 2017, arXiv:1707.01155.
2. SAHU A K, LI T, SANJABI M, et al. Federated optimization for heterogeneous networks[J]. arXiv preprint, 2018, arXiv:1812.06127.
3. KONEČNÝ J, MCMAHAN H B, YU F X, et al. Federated learning: strategies for improving communication efficiency[J]. arXiv preprint, 2016, arXiv:1610.05492.
4. CALDAS S, KONEČNY J, MCMAHAN H B, et al. Expanding the reach of federated learning by reducing client resource requirements[J]. arXiv preprint, 2018, arXiv:1812.07210.
5. NISHIO T, YONETANI R. Client selection for federated learning with heterogeneous resources in mobile edge[C]//ICC 2019-2019 IEEE International Conference on Communications. Piscataway: IEEE Press, 2019: 1-7.
6. YOSHIDA N, NISHIO T, MORIKURA M, et al. Hybrid-FL for wireless networks: cooperative learning mechanism using non-IID data[J]. arXiv preprint, 2019, arXiv:1905.07210.
7. LI T, SAHU A K, TALWALKAR A, et al. Federated learning: challenges, methods, and future directions[J]. IEEE Signal Processing Magazine, 2020, 37(3): 50-60.

YANG Q, LIU Y, CHEN T, et al. Federated machine learning: concept and applications[J]. ACM Transactions on Intelligent Systems and Technology, 2019, 10(2): 1-19.