分层强化机制对边缘学习的激励驱动长期优化

Incentive-Driven Long-term Optimization for Edge Learning by Hierarchical Reinforcement Mechanism

Introduction

边缘学习同云端的分布式机器学习相似，利用本地数据对全局模型进行训练。近年来，许多学者针对边缘学习进行了研究，包括节点选择，资源调度，节点攻击等，但这些工作都是基于节点自愿参与到模型训练当中。

文中指出，由于边缘学习需要节点投入资源(电力，数据等)，因此需要一种奖励机制使得边缘节点弥补资源等消耗。

相关学者对边缘学习的激励机制进行了探索，但是它们*1)*没有考虑长期可持续性，只对单轮训练进行了优化，这将导致预算的快速消耗； *2)*另外没有对各种模型特点进行考虑，这将无法保证模型最终的准确性。

文章将边缘学习算法的性能加入到激励机制的设计中，针对参数服务器无法获得边缘节点的问题，提出采用经验驱动的DRL方法优化激励机制；同时面对DRL模型无法适应长短期的双层优化目标，提出了分层的DRL方法Chrion，从而优化边缘节点的定价策略。

本文贡献有三：将学习算法的性能加入到激励机制的考量中；提出了一种分层的DRL方法；基于Pytorch平台验证了该机制的有效性。

Background

- 边缘学习

边缘学习和FL类似，训练数据由边缘设备产生，在 parameter server上进行梯度汇聚。

- 深度强化学习

深度强化学习提供一个代理，根据环境动作状态的转换获取奖励，通过学习环境的状态转换关系，使得总体回报最大。这种决策选择基于马尔可夫决策树。由于决策树无法涵盖所有的决策可能，因此往往采用函数近似技术表示策略。本文采用DNN对Chiron进行表示。

System Model

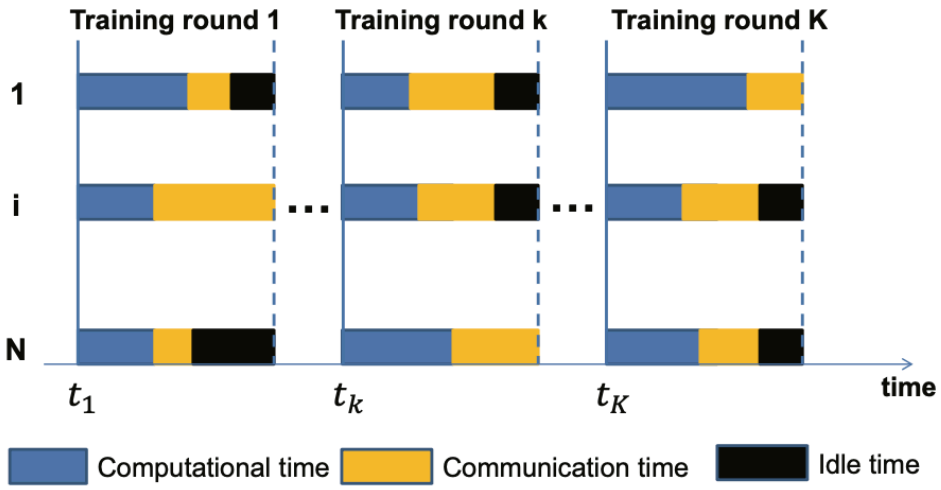


Figure 1. Illustration of edge learning operational process.

文章采用全同步随机梯度下降的思想构建模型。并将从时间片的角度对训练时间进行分析：

$$T_{i,k}^{cmp} = \frac{\sigma c_i d_i}{\zeta_{i,k}}$$

得到时间花费如上，其中σ代表本地迭代次数，c_i代表单位训练数据需要的CPU时钟数；d_i表示训练数据总量；$\zeta_{i,k}$表示CPU时钟频率。传输花费有：

$$T_{i,k}^{com} = \frac{\xi}{B_{i,k}}$$

其中ξ为模型大小，$B_{i,k}$为边缘节点带宽。因此第k次运算的时间花费总量为$T_k = \max\left(T_{i,k}^{cmp} + T_{i,k}^{com}\right)$。文章按照广泛使用的能量模型，得到计算能量消耗为$E_{i,k}^{cmp} = \sigma\alpha_i c_i d_i \zeta_{i,k}^2$，其中$\alpha_i$为边缘节点计算芯片的有效电容系数；通讯消耗为$E_{i,k}^{com} = \epsilon_i T_{i,k}^{com}$，其中$\epsilon_i$为网络通讯时产生的单位能量消耗，与CPU时钟频率$\zeta_{i,k}$相互独立，因此能量消耗为$E_{i,k} = E_{i,k}^{cop} + E_{i,k}^{com}$。

假设CPU单位频率提供的价格为$p_{i,k}$，则边缘节点的实用收益评估为$u_{i,k} = p_{i,k}\zeta_{i,k} - E_{i,k}$。对于边缘学习任务来说，其目标为快速达到预期模型准确度，因此其实用收益评估为

$$u = \lambda A(w_K) - \sum_{k=1}^{K} T_k$$

其中λ为自定义参数指标，$A(w_K)$表示模型K次训练后的准确率。这样文章将算法性能同激励机制结合起来，从而保证模型准确率的同时吸引更多节点参与计算。

Problem Formulation And Analysis

- 问题制定

在进行边缘计算时，边缘节点会同parameter server进行定价协同，对于边缘节点来说，其收益可以表示为：

$$OP_{i,k} : max\ u_{i,k}$$
$$s.t.\ \zeta_{i,k} \in [\zeta_i^{min}, \zeta_i^{max}]$$
$$u_{i,k} \geq \mu_i$$

其中μ_i为边缘节点参与计算的基础收益。对于ps节点，其目标为：

$$OP_{PS}: max\ u,$$
$$s.t. \sum_{k=1}^{K}\sum_{i=1}^{N} p_{i,k}\zeta_{i,k} \leq \eta$$

其中η为ps节点预算。

- 优化策略分析

对于边缘节点，文章通过数学分析指出最优策略为$\zeta_{i,k}^{*} = \frac{p_{i,k}}{2\sigma\alpha_i c_i d_i}$，此时边缘节点计算时间为$t_{i,k}^{cmp,*} = \frac{2\alpha_i\sigma^2 c_i^2 d_i^2}{p_{i,k}}$；对于PS节点，由于一些参数无法从边缘节点获取，因此无法使用数学推导的方法获得最佳结果，但通过边缘节点的训练时间来反馈激励策略的表现。

Hierarchical Reinforcement Machanism Design

以上PS节点的优化问题虽然已经提出，但是由于1）边缘节点参数无法获取和2）由于DNNs，$A(w_K)$无法通过数学模型构建，因此文章提出采用DRL方法进行策略优化，减少边缘节点的空闲时间。

传统的DRL往往是单一代理，这种方式并不能很好的解决边缘学习中遇到的问题，因此提出分层DRL，从而达到长短期目标：1）长期目标：合理分配每轮定价，从而增加迭代轮数；2）短期目标：每轮定价固定后，合理分配每个边缘节点的定价从而使训练时长一致。

文章提出了两层DRL：Exterior Agent和Inner Agent来分别解决上述两个目标，并使用PPO做为分层强化算法。

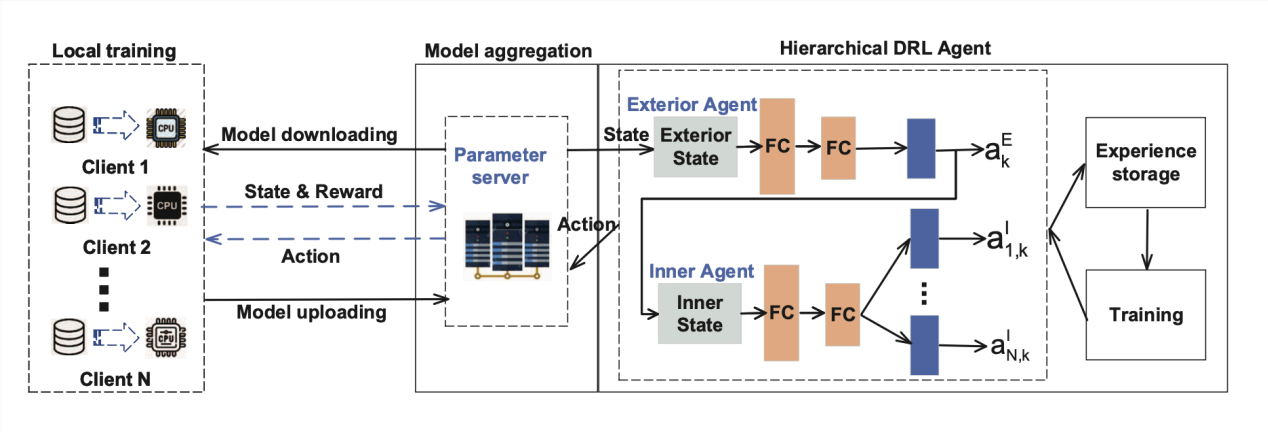


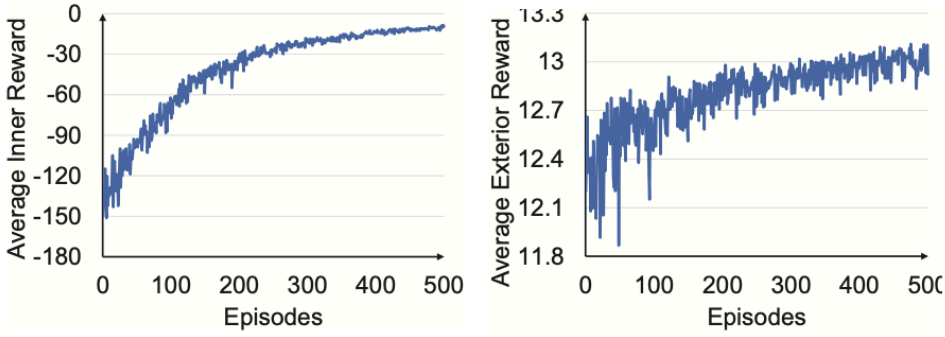
Figure 2. The architecture of Chiron.

其中，两者的回报分别设置为：

$$Exterior\ Reword:\ r_k^E = \lambda(A(w_k) - A(w_{k-1})) - \lambda T_k$$
$$Inner\ Reword:\ r_k^I = -\sum_{i=1}^{N}(T_k - T_{i,k})$$

Performance Evaluation

文章在Mnist&&Fashion Mnist和CIFAR-10两个数据集上进行实验验证。

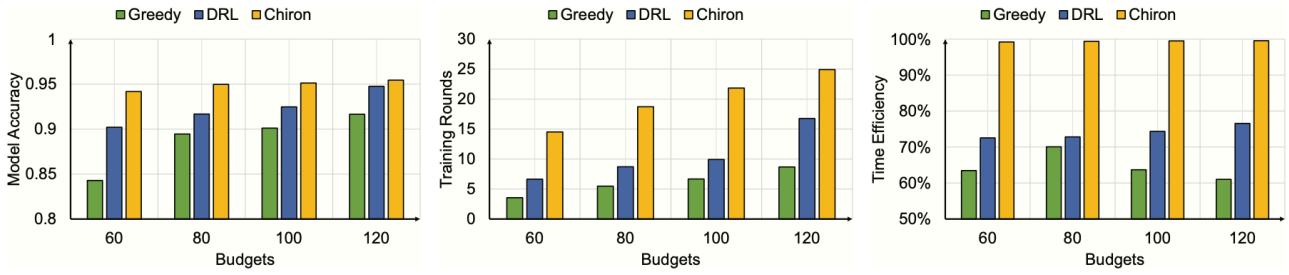


(a) Inner DRL agent. (b) Exterior DRL agent.

Figure 3. Convergence of Chiron under MNIST.

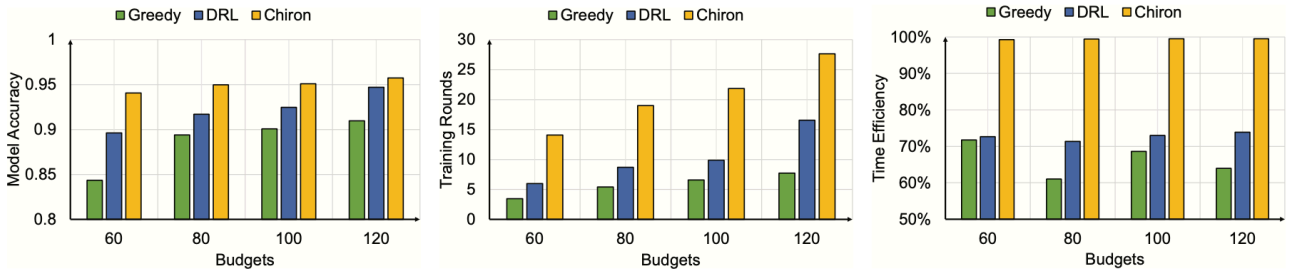
在Mnist数据集上，文章展示出随着迭代次数的增加，节点能够稳定熟练至较高的回报。

同时，文章在三个数据集上与Greedy(随机产生行为，从中选取收益最大的执行)和DRL-based(仅考虑单次迭代的能量损耗和时间花费)。



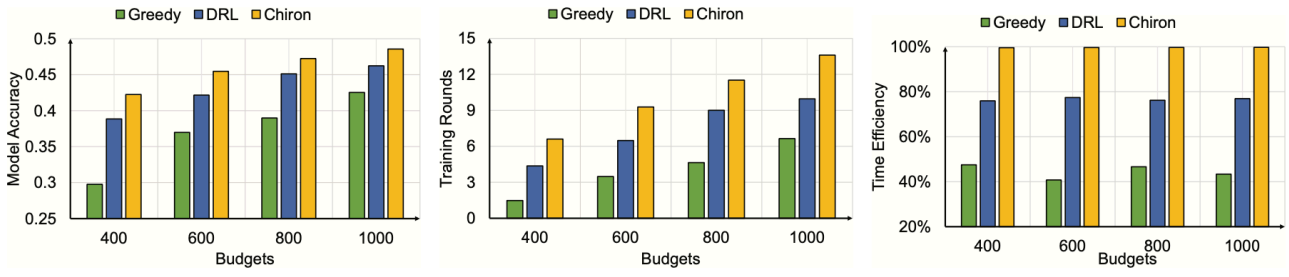
(a) Global model accuracy. (b) Training rounds. (c) Time efficiency.

Figure 4. Performance under MNIST when varying the total budgets.



(a) Global model accuracy. (b) Training rounds. (c) Time efficiency.

Figure 5. Performance under Fashion-MNIST when varying the total budgets.



(a) Global model accuracy. (b) Training rounds. (c) Time efficiency.

Figure 6. Performance under CIFAR-10 when varying the total budgets.

实现显示了Chiron能够更好的利用节点资源并进行更多迭代从而获取更好的模型准确率。