FedRL-Prox: a federated learning hyperparameters optimization via reinforcement learning

Yushen (Ido) Chen YCHE2692@UWO.CA
Department of Computer Science
University of Western Ontario
Ontario, Canada

Zifan Zhu ZZHU459@UWO.CA
Department of Computer Science
University of Western Ontario
Ontario, Canada

Min Zhang MZHAN782@UWO.CA
Department of Computer Science
University of Western Ontario
Ontario, Canada

Editor: Yushen (Ido) Chen, Zifan Zhu, Min Zhang

Abstract

Federated Learning (FL) is a distributed learning paradigm that specifically addresses privacy concerns and data security of traditional centralized learning. Since FL requires each training participant to upload their training updates rather than the actual data, system and statistical heterogeneity across devices significantly impacts the prediction accuracy and robustness of the trained model. Thus, Federated Proximal (FedProx), as one of the most popular FL optimization algorithms proposed by Tian et al. in 2019, is introduced to alleviate the heterogeneity problem within the FL network by introducing a model penalty to reduce the deviation between client local and server global model. In our work, we aim to develop a schema to adaptively control the strength of the model penalty on each client's model updates. We propose a novel approach that leverages reinforcement learning (RL) techniques, particularly the Proximal Policy Optimization (PPO) algorithm, to automatically adjust the hyperparameter μ value throughout the FL training process. Our goal is to improve the overall accuracy and convergence speed of the FL model by continuously refining the μ value for each client using an RL agent. Our methodology involves: iterative global model updates, local training rounds, and RL-based μ selection.

Keywords: Federated Learning, Hyper Parameter Optimization, Reinforcement Learning, Federated Proximal

1 introduction

Federated learning has been proposed for training machine learning models across decentralized devices to preserve data privacy. However, due to the nature of this new framework, the model updates from different clients may be generated from different datasets, computational capabilities, and other system configurations. All these characteristics can impact the prediction rate and convergence of the trained model.

Considering this challenge, one solution to address the heterogeneity problem is to use FedProx algorithm, which introduces penalty terms to the optimization objective on local model updates. For example, we can use the μ parameter in the FedProx algorithm to control how much we want to penalize the local model updates. It is not easy to select an appropriate μ value for all clients, as the data heterogeneity is not known before the training process. Therefore, the algorithm brings a new challenge to the Federated Learning system.

The primary motivation of this work is to address the new challenge of selecting the μ value. We decided to use reinforcement learning to dynamically adjust the μ value during the training process.

Our experiment used FL for image classification. We preprocessed the data to simulate the independent and identically distributed (iid) data and non-independent and identically distributed (non-iid) data. Based on the data, we tested the performance of the FedAvg algorithm and the FedProx algorithm on each datasets. Then for the FedProx algorithm, we used the PPO algorithm to adjust the μ value during the federated learning process. We measured the accuracy, and convergence speed of each training approach.

The contribution of our project is to utilize FL on the image classification problem while using the FedProx algorithm with RL to optimize hyperparameter μ. Although we only tested the method on image classification, the method can be applied to any other FL problem using the FedProx algorithm.

The rest of this report is organized as follows: Section II describes the Background and Related work. Section III shows the methods of our research. The proposed approach and the data preparation are introduced.Section IV presents the results of our research. The last section is our conclusion and future work.

2 Background and Related Work

We briefly reviewed studies related to hyperparameter optimization, federated learning, and federated learning optimization algorithms.

2.1 Secure data privacy

Through some studies, it is not hard to see the performance advantage of deep learning networks in many identification, and prediction problems. However, data privacy has always been an issue that most centralized training methods cannot resolve. Therefore, FL has been proposed to address this issue. FL allows decentralized training to protect data privacy. It does not require centralized data storage, the only thing that needs to be sent to the centralized server is model updates. This characteristic makes FL a good choice for privacy protection. In our research, we choose FedProx as the Federated Learning optimization algorithm for handling the heterogeneity problem in the FL network. The algorithm will be introduced in the following subsection.

2.2 Federated Learning Optimization Algorithm

At first, McMahan et al. (2016) [1] proposed a FedAvg algorithm to train deep learning model under the federated learning network. The algorithm is effective, simple, and robust. The basic training process can be described as follows: for each global training round, the central server sends the global model to all clients for local training. Then the trained local model updates can be sent to a centralized server. The central server aggregates the updates into the global model. It's good for handling iid data. However, when dealing with the non-iid data, the performance of the model might be affected due to the heterogeneity of the data. So, Li et al. (2018) [2] proposed FedProx algorithm that can handle such cases better than FedAvg (address the challenge of data or system heterogeneity). We currently only focus on the data heterogeneity. This algorithm introduces a proximal term to regularize local model updates. The objective function of FedProx can be described as follows:

$$\min_{w} h_k(w; w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|_2^2 \tag{1}$$

In this formula, $F_k(w)$ is the client objective function, the second term is the model penalization times the penalization constant $\mu/2$ in our project, we only focus on the selection of μ, which is the hyperparameter to control client local model updates.

2.3 Hyperparameter Optimization

There are many hyperparameter optimization approaches in the literature, To our best knowledge, they are all designed for traditional centralized machine learning (ML) or deep learning (DL) models and none of them can take effect without violating the privacy concerns within FL setting. Moreover, since FL involves local devices to perform model training, even the same scenario (same dataset and models) can significantly vary in training results with different sets of training devices, which makes traditional hyperparameter optimization approaches non-generalizable. Some related works include an RL-based framework to improve the DL hyperparameter selection for visual object tracking introduced by Dong et al. (2021) [3]. Wu et al.(2019) [4] introduced an RL architecture, namely, RPR-BP to optimize the hyperparameter selection in any ML model on a given dataset. Chen et al. (2019) [5] present a method to minimize the deployment cost of RL-based hyperparameter selection by predicting the reward signal rather than computing it. Huang et al. (2021) [6] proposed an RL-based hyperparameter optimization framework to enable auto-tuning in self-supervised robotic learning. After the research of these works, we found that the RL-based FedProx hyperparameter optimization is still a novel approach in the FL domain. Therefore, we decided to design a reinforcement learning framework, namely FedRL-Prox, to adaptively optimize hyperparameters selection in FedProx. Specifically, the RL agent selects the value of μ based on the reward signal from the cross-validation results after several rounds of FL.

3 Methods

In this section, we discuss the method of our research, including the research objectives and research methodology. For the research methodology, we divide it into four parts: the proposed approach, dataset and preparation, experiment implementation, and the simplified FedProx.

3.1 Research Objective

We have two research objectives to accomplish: (a) optimizing accuracy and convergence speed through RL, and (b) ensuring privacy protection with RL-driven parameter adjustment.

(a) Our primary goal is to exploit the capabilities of RL to dynamically adjust parameters, thereby improving accuracy and accelerating convergence rates in various domains. By applying RL techniques, we aim to create adaptive systems that can autonomously adjust their parameters based on real-time feedback, leading to improved performance and faster convergence in complex environments.

(b) The rapid convergence facilitated by RL-based parameter tuning results in fewer computational iterations and reduced communication overhead. This efficiency not only accelerates model training, but also minimizes the exchange of sensitive information between devices and centralized servers. With fewer rounds of communication and computation, the exposure of sensitive data to potential adversaries is significantly reduced. In this way, fast convergence becomes a critical factor in enhancing privacy by reducing the overall surface area for potential privacy breaches.

3.2 Proposed Approach

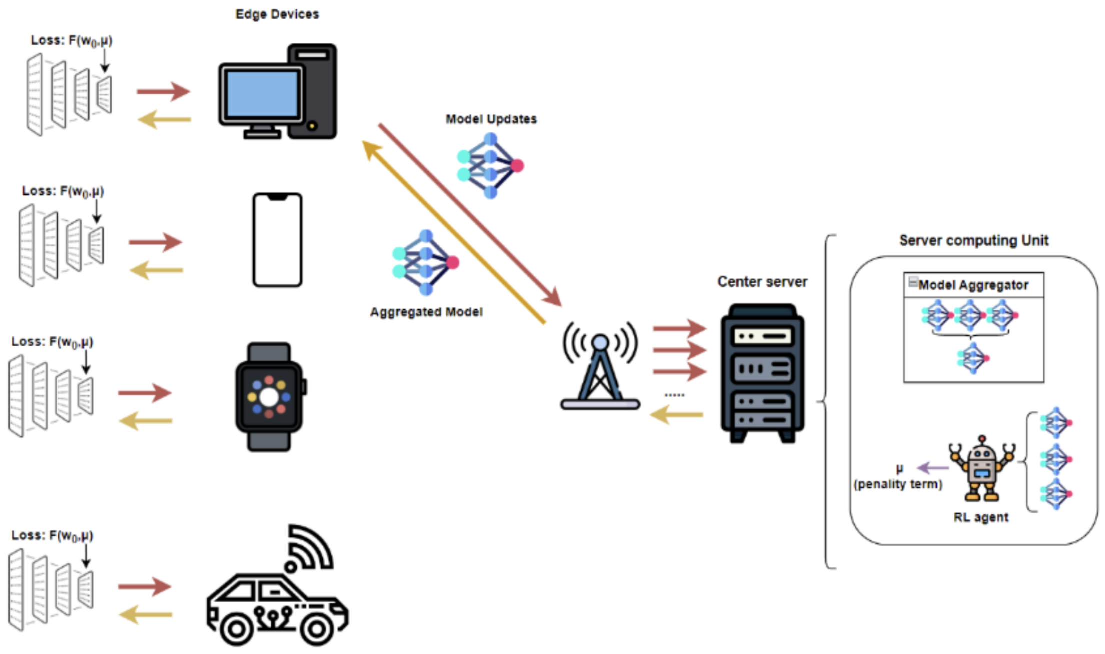

Figure 1: System Model

Fig. 1 illustrates our architecture.

First, a diverse set of devices is deployed, including computers, smartphones, smartwatches, and cars, each equipped with local data and computational capabilities. Each device trains a local model, denoted as $F(w, \mu)$, where w represents the model weights and μ represents an adjustable parameter.

Then, local models on edge devices send their updates to a central server for aggregation, with arrows illustrating the flow of these updates to the central server.

The central server merges the model updates received from the edge devices to create a global or aggregated model. This aggregated model encapsulates the collective knowledge of all participating devices. The server orchestrates the federated learning process by aggregating model updates from edge devices and redistributing the aggregated model back to them. A dedicated component on the server oversees the consolidation of model updates from edge devices.

The approach focuses on using RL to improve model accuracy and speed while maintaining privacy. RL dynamically adjusts model settings, improving accuracy and speeding learning across domains. RL also helps reach the best settings quickly, reducing the risk of exposing sensitive information during communication between devices and servers. In experiments, we use RL alongside GPU acceleration and federated learning techniques such as Proximal Policy Optimization (PPO) to efficiently train models.

3.3 Dataset and Preparation

We chose the CIFAR-10 dataset of 50,000 data points as the basis for our research efforts. Our goal was to improve model performance and efficiency through the use of innovative techniques.

To facilitate comprehensive analysis, we divided our dataset into two subsets: one with independent and identically distributed (iid) data, and the other with non-id characteristics. While managing the iid data proved relatively straightforward due to its consistent quality, addressing the non-id data distribution posed significant challenges. Our strategy involved simulating non-id data distribution by distributing limited classes to individual clients.

3.4 Experimental Implementation

In our implementation, we take advantage of GPU acceleration within the PyTorch framework to speed up model training. Specifically, we design convolutional neural networks (CNNs) tailored to the classification task, ensuring uniformity across clients participating in the federated learning process. Each client uses the same CNN architecture for training, maintaining consistency and facilitating seamless model aggregation.

To optimize resource utilization, including GPU allocation, we use a single-client approach where GPU and other computational resources are allocated to one client at a time. This resource allocation strategy ensures efficient use of hardware resources without unnecessary contention.

In our federated learning setup, we use the Proximal Policy Optimization (PPO) algorithm implemented using the stable_baseline3 library. We configure the PPO algorithm with specific hyperparameters, setting the number of steps (n_steps) to 20 and the total number of training time steps to 1000. We apply RL techniques, specifically adjusting the mu parameter, to continuously improve performance. Our experiments include training on five clients using non-IID datasets, ensuring diverse data inputs and robust model generalization.

3.5 Simplified FedProx

In our model, we adopted a simplified FedProx algorithm. Departing from the conventional FedProx method, we streamlined our approach for ease of comparison. We used 10 sets of non-iid data and iid data, all derived from the CIFAR-10 dataset.

To provide a comprehensive analysis, we ran experiments with dynamic values of μ, ranging from 0.01 to 9. This wide range of μ values allowed us to explore different levels of regularization and their impact on model performance. In each experiment, the model was trained using the corresponding μ value, thus covering a wide range of regularization strengths.

4 Experimental Results

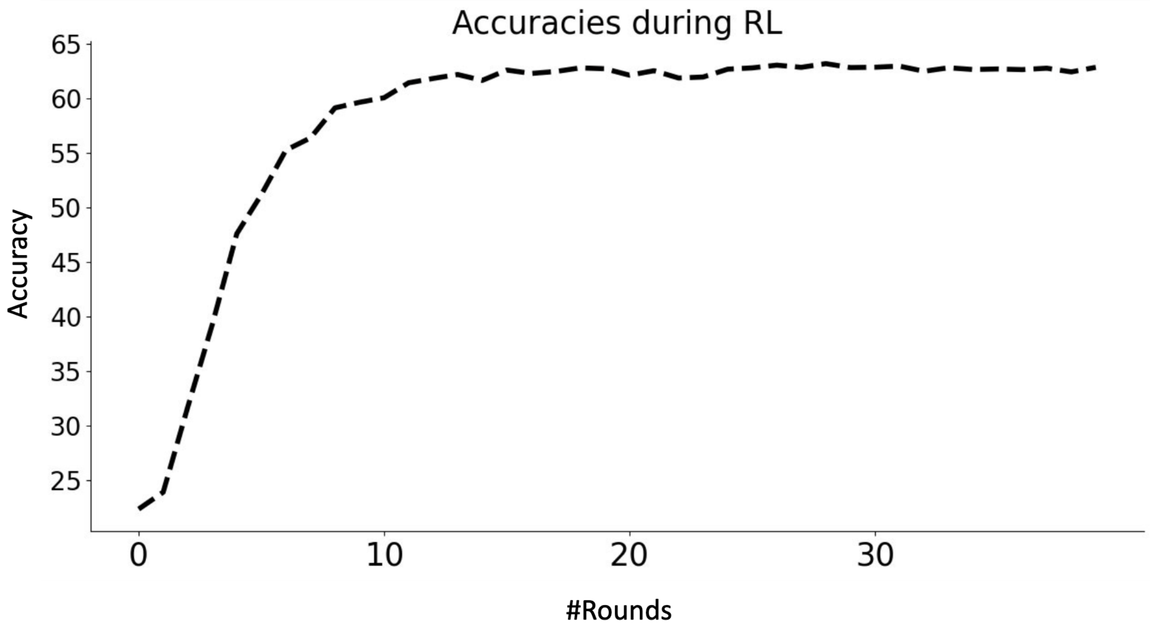

Figure 2: Progression of Accuracy in Reinforcement Learning.

Fig. 2 serves as a concise visual summary of our latest research findings. One of the key observations is the significant improvement in accuracy, which has jumped to approximately 65%. This remarkable achievement represents a remarkable 3% improvement over the performance of the standard model. Another is that we observed a significant benefit in terms of convergence speed. The graph clearly shows a significant reduction in the time required for our model to converge to optimal performance compared to previous iterations.

Our experimental journey was not without its challenges, particularly with regard to overfitting, a common problem in machine learning efforts. To mitigate this, we implemented rigorous measures to address overfitting and ensure that the performance of our model remained robust and reliable. In addition, maintaining consistent loss profiles across different rounds of experiments was critical to measuring the effectiveness of our interventions and improvements.

An important aspect of our methodology was the implementation of continuous data randomization. By introducing variability into the training data, we aimed to improve the generalization capabilities of our model, thereby enhancing its real-world applicability and performance.

Therefore, we can use RL to optimize FL.

5 Conclusion and Future Work

In our study, we used RL to fine-tune the Federated Proximal algorithm. When compared to the original FL approach, our FedRL-Prox model showed notable performance gains, with an improvement of approximately 3%. Furthermore, FedRL-Prox showed a slightly accelerated convergence speed compared to some μ variants. This research highlights a promising avenue for the optimization of FL methods, providing a new perspective in the field. Going forward, our goal is to explore a broader range of parameters to potentially achieve even more remarkable results. In addition, we advocate the exploration of alternative RL models to further enrich our understanding and increase the effectiveness of federated optimization techniques.

References

[1] H. B. McMahan, E. Moore, D. Ramage, S. Hampson, B. Agüera y Arcas, “Communication-Efficient Learning of Deep Networks from Decentralized Data,” in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS) 2017*, Journal of Machine Learning Research: Workshop and Conference Proceedings, vol. 54, 2017.

[2] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, V. Smith, “Federated Optimization in Heterogeneous Networks,” in *Proceedings of MLSys 2020.*

[3] X. Dong, J. Shen, W. Wang, L. Shao, H. Ling and F. Porikli, ”Dynamical Hyperparameter Optimization via Deep Reinforcement Learning in Tracking,” in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, 2021, pp. 1515-1529.

[4] J. WU, S. CHEN and X. CHEN, ”RPR-BP: A Deep Reinforcement Learning Method for Automatic Hyperparameter Optimization,” in *International Joint Conference on Neural Networks (IJCNN)*, Budapest, Hungary, 2019, pp. 1-8.

[5] S. Chen, J. Wu and X. Chen, ”Deep Reinforcement Learning with Model-Based Acceleration for Hyperparameter Optimization,” in *IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, Portland, OR, USA, 2019, pp. 170-177.

[6] J. Huang, J. Rojas, M. Zimmer, H. Wu, Y. Guan and P. Weng, ”Hyperparameter Auto-Tuning in Self-Supervised Robotic Learning,” in *IEEE Robotics and Automation Letters*, vol. 6, 2021, pp. 3537-3544.