UCBFed: Using Reinforcement Learning Method to Tackle the Federated Optimization Problem

Wanqi Chen, Xin Zhou

UCBFed: Using Reinforcement Learning Method to Tackle the Federated Optimization Problem

Wanqi Chen[1,2][0000−0002−4655−5381] and Xin Zhou[1,2][0000−0002−8233−4412]

[1] Institute of Software, Chinese Academy of Sciences, Beijing, China
[2] University of Chinese Academy of Sciences, Beijing, China
{wanqi2020,zhouxin}@iscas.ac.cn

Abstract. Federated learning is a novel research area of AI technology that focus on distributed training and privacy preservation. Current federated optimization algorithms face serious challenge in the aspects of speed and accuracy, especially in non-i.i.d scenario. In this work, we propose UCBFed, a federated optimization algorithm that uses the Upper Confidence Bound(UCB) method to heuristically select participating clients in each round's optimization process. We evaluate our algorithm in multiple federated distributed datasets. Comparing to most widely-used FedAvg and FedOpt, the UCBFed we proposed is superior in both the final accuracy and communication efficiency.

Keywords: Federated Learning · Upper Confidence Bound · Distributed Learning · Optimization.

1 Introduction

Federated learning is an emerging field in machine learning research. In the federated setting, training data remained distributed in a large number and variety of edge devices, such as computers, smart phones and many other IOT devices. Data in those devices might be useful for training a global model to accomplish specific missions, such as image classification, object detection and others. However, due to privacy, liability, cost and some other issues, its not allowed to move this data together in the federated setting.

There are many open challenges in federated learning research, such as increasing communication-efficiency[9], preserving users privacy[3] and defensing against malicious attacks[13]. In this work we address the speed and accuracy of federated optimization, especially in heterogeneous(non-i.i.d) situation.

In this paper, we propose a new algorithm called UCBFed. Instead of selecting clients randomly in each round in polular method like FedAvg[10], UCBFed uses the Upper Confidence Bound(UCB) method[1] which is originally designed for multi-armed bandit problem in Reinforcement Learning. The UCB algorithm take both history rewards and number of times that one client has been chosen into account to determine which client will be chosen in a specific round. To define the reward, referring to the experience of FedReID[14], we use cosine distance of the last layer of backbone network between old global model and

newly trained client model as a metric. We also use this metric as the weight when aggregate models from clients, replacing the number of data cases in each clients used by FedAvg algorithm. The remainder of this paper is organized as follows. In Section 2, we provide an simple overview of related work. In Section 3 we propose our UCBFed algorithm and describe two main components of the algorithm. In Section 4 the comparison among baseline algorithms and ours is presented. Finally, we expound our conclusion and future work that need to continue in Section 5.

2 Related Work

2.1 Federated Learning

The very beginning of the federated learning research is from 2015, with the first publications on federated averaging(FedAvg). There are many research directions in this area, including improving accuracy and speed of federated optimization, reducing the communication burden during the federated process and protect users privacy in federated settings. For the first direction, many algorithms including FedProx[8] and SCAFFOLD[6] use information from central model to correct the local updates. Then, methods such as model compression and model pruning are applied to the reduction of communication cost. Finally, Bagdasaryan[2], Wang[11] and Zhu[13] investigate many aspects about how to attack and defense of the federated learning.
There are different scenarios of federated learning. In horizontal federated learning, different clients have data that have same features but different IDs. On the contrary, in vertical scenario, different sets of data have same IDs but different features. And in transfer federated learning, both features and IDs are different.

Horizontal Federated Learning In this paper, our work focus on horizontal federated learning. The problem can be formulated as follows:
Consider a scenario with C clients. The i^{th} client has D_i samples. Let z_i be the data domain of each client and $L(w, z_i)$ be the likelihood function of i^{th} client, then:

$$\arg\max_w f(w) = \sum_{i=1}^{C} \frac{D_i}{\sum_{i=1}^{C} D_i} L(w, z_i)$$

FedAvg In FedAvg algorithm, each client train local model using local data in muliple batches and upload the gradients to the server. Then the server will weight averaging all the gradients using local data size as the weight.Then server will transfer back the weights to client and start next rounds of training. Though FedAvg is successful in several fields, it still has some space for improvement. As our work concerns, the FedAvg selects clients randomly in each round, ignoring the quality of each client's data. Unavoidably it will harm the result of optimization. Therefore our work present a strategy to choose client in each round.

2.2 Multi-armed Bandit Problem

The multi-armed bandit problem can be described as follows:
The multi-armed bandit has $K \in N^+$ levers and the reward delivered by each bandit is associated with a distribution $R_i \in B = \{R_1, ..., R_K\}$. The player repeatedly plays one lever per round and observes the associated reward. The aim is to maximize the sum of rewards collected by players.
To solve the multi-armed bandit problem, an algorithm should tackle with a trade-off between two choices: exploit the levers that have better history rewards or explore those levers that are rarely visited.
Several algorithms have been proposed, including ϵ-greedy algorithm, Thompson sampling, Upper Confidence Bounds(UCB) and more.

Upper Confidence Bound(UCB) Algorithm In UCB algorithm, each lever a has a metric A_t given by:

$$A_t = \arg\max_a [\frac{H_t(a)}{N_t(a)} + c\sqrt{\frac{\log t}{N_t(a)}}]$$

$H_t(a)$ is the sum of history rewards of the lever at time step t and $N_t(a)$ is the number of times that the level has been selected before. Finally, c is a control variable that controls the level of exploration.
During each round t, UCB algorithm plays the lever that has the maximum value of A_t and updates A_t of all levers.

3 The proposed UCBFed Algorithm

3.1 Cosine Distance Weight(CDW)

To apply UCB thought to the process of choosing participating clients, we need a metric to define the "quality" of each client's training result. A simple yet effective idea is to use the amount of information or the change that a single client contributes to the model. Hence, inspired by the work of FedReID[14], We choose Cosine Distance Weight(CDW) as the metric. CDW measures the information that a client has learned by the cosine distance between the outputs of feature extraction parts of the model before and after client training process.

3.2 Reward Allocation

Once a round of client training process is finished and the server gets all participating clients' CDWs, we can allocate actual rewards to these clients based on the metrics. UCBFed gives those clients which have the largest CDW a positive reward(+1) while other clients can only get zero rewards.

Algorithm 1: UCBFed. S is the set of all clients. K is the number of clients that participate in a round; B is the local minibatch size, E is the number of local epochs, and η is the learning rate.D is data size of all clients

Server executes:
  Initialize w_0
  Initialize: $\forall c_i \in S, H_t(c_i) \leftarrow 0, N_t(c_i) \leftarrow 0$
  for *each round t=1,2...,T* **do**
    for $c_i \in S$ **do**
      $A_t(c_i) = \frac{H_t(c_i)}{N_t(c_i)} + \beta\sqrt{\frac{\log t}{N_t(c_i)}}$
    end
    $S_t \leftarrow$ clients that have top K largest A_t
    for *each client* $c_k \in S_t$ ***in parallel*** **do**
      $w_{t+1}^k, CDW_{t+1}^k \leftarrow ClientUpdate(k, w_t)$ $N_{t+1}(c_k) \leftarrow N_t(c_k) + 1$
    end
    update H_t using Algorithm 3
    $w_{t+1} \leftarrow \sum_{k=1}^{K} \frac{CDW_{t+1}^k}{\sum_{j=1}^{K} CDW_{t+1}^j} w_{t+1}^k$
  end
ClientUpdate:
  for *each local epoch* **do**
    for *batch* $b \in$ *training data* **do**
      $w \leftarrow w - \eta \nabla l(w; b)$
    end
  end
  $CDW_k \leftarrow$ computing CDW using Algorithm 2
  return CDW_k to server

3.3 Clients Selection and Model Aggregating

After we get reward and CDWs of participating clients, we can start to aggregate the client models into a new global model and select the participating clients in the next round. The algorithm is generally based on FedAvg but has two improvements. First, instead of random selecting clients in each round, we choose the clients that have the largest Upper-Confidence-Bound value A_t; Second, instead of using data size as the weight of weight averaging process of model aggregating, we use a CDW-based weight.
The full process of the UCB algorithm is in Algorithm 1.

4 Experiments

We evaluate our **UCBFed** algorithm in multiple datasets. We compared our algorithm with the most widely used algorithm–**FedAvg** and **FedOpt**. UCBFed has shown promising results in various datasets.

4.1 Experiments Details

Datasets and setup We run experiments on three different datasets: EMNIST[4], SHAKESPEARE[8] and CIFAR-100[7]. These datasets have been already used in different works in these areas before. **EMNIST** is an handwritten character classification datasets. It's collected from 3400 different writers; **Shakespeare** dataset is a next-word prediction task on *The Complete Works of William Shakespeare* and each speaking role corresponds to a device; Finally, **CIFAR100** is a tiny image classification dataset consists of 100 classes of 60000 samples. And for the federated setting experiment it is divided into 500 parts and each part corresponds to a device. Statistic are summarized in Table 1.

Table 1. Summaries of Models using in different tasks

Dataset	Device	Training Samples	Testing Samples
Shakespeare	143	413629	103477
CIFAR-100	500	50000	10000
EMNIST	3400	671585	77483

Federated Models In this experiments we train different models in different datasets in federated settings using UCBFed and FedAvg. More specifically, we use a simple deep convolution network on **EMNIST** dataset and a simple LSTM network on **Shakespeare** dataset. A ResNet-18 network is adopted on CIFAR-100 mission to represent the scenario that a complicated network is trained using federated settings. In federated scenarios, computing resources of client devices are usually limited, hence the batch-size might be very small. In this case, traditional Batch Normalization may lose its efficiencies.So we use batch-size-irrelevant Group Normalization[12] instead of Batch Normalization in ResNet-18 network.

Implementation We implement our experiments code based on FedML framework.[5] More specifically, we use the single-machine stimulation environment the framework provided to demonstrate our experiment. Based on FedAvg codes, we modify the process of client sampling and weight averaging. For all tasks, we set $\alpha = 0.8$ and $E = 5$. Other hyperparameters are summarized in Table 2.

Metrics For each task, we evaluate our algorithms using two metrics. First is the highest accuracy that an algorithm can achieve. Another commonly used metric in federated optimization area is the communication cost for reaching a specific accuracy target. In this experiment, the communication cost of each round is equal between UCBFed and FedAvg. Hence, we use the communication round before reaching a specific accuracy target as an equivalent.

Table 2. Summaries of hyperparameters

Dataset	Learning Rate	Device per Round	Total Round
Shakespeare	0.8	10	200
CIFAR-100	0.001	10	1000
EMNIST	0.03	50	1000

4.2 Experiment Results

As it has shown in Table 3 and Table 4, UCBFed is superior than FedAvg in all three tasks. Comparing to FedOpt, UCBFed has achieved better highest accuracy. However, in the aspect of communication round, UCBFed takes more round to achieve the target accuracy than FedOpt in EMNIST and CIFAR-100.

Table 3. Highest Accuracy

Dataset	UCBFed(ours)	FedAvg	FedOpt
Shakespeare	**0.519**	0.513	0.386
EMNIST	**0.851**	0.849	0.834
CIFAR-100	**0.264**	0.235	0.224

Table 4. communication round of reaching a specific accuracy target

Dataset(Target Accuracy)	UCBFed(ours)	FedAvg	FedOpt
Shakespeare(0.5)	**35**	115	N/A
EMNIST(0.8)	55	60	**50**
CIFAR-100(0.2)	185	445	**130**

5 Conclusion and Future Works

In this work, we have proposed UCBFed, a federated learning algorithm that tackles the speed and accuracy of federated optimization. UCBFed uses the Upper Confidence Bound(UCB) method to select the participating clients of each round to improve the quality of optimization. In our experiment UCBFed shows significantly better result than FedAvg. It indicates that UCBFed improves the convergence behavior of federated learning in realistic heterogeneous network.
The future work includes the application to various and complicated federated scenarios, including federated object detection and ReId. Then, it is interesting to find if there are better metrics to replace CDW. Finally, the combination of UCBFed and model personalization is also an interesting research point.

References

1. Auer, P.: Using upper confidence bounds for online learning. In: Proceedings 41st Annual Symposium on Foundations of Computer Science. pp. 270–279. IEEE (2000)
2. Bagdasaryan, E., Veit, A., Hua, Y., Estrin, D., Shmatikov, V.: How to backdoor federated learning. In: International Conference on Artificial Intelligence and Statistics. pp. 2938–2948. PMLR (2020)
3. Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H.B., Patel, S., Ramage, D., Segal, A., Seth, K.: Practical secure aggregation for privacy-preserving machine learning. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security. pp. 1175–1191 (2017)
4. Cohen, G., Afshar, S., Tapson, J., Van Schaik, A.: Emnist: Extending mnist to handwritten letters. In: 2017 International Joint Conference on Neural Networks (IJCNN). pp. 2921–2926. IEEE (2017)
5. He, C., Li, S., So, J., Zhang, M., Wang, H., Wang, X., Vepakomma, P., Singh, A., Qiu, H., Shen, L., et al.: Fedml: A research library and benchmark for federated machine learning. arXiv preprint arXiv:2007.13518 (2020)
6. Karimireddy, S.P., Kale, S., Mohri, M., Reddi, S.J., Stich, S.U., Suresh, A.T.: Scaffold: Stochastic controlled averaging for on-device federated learning. arXiv preprint arXiv:1910.06378 (2019)
7. Krizhevsky, A., Hinton, G., et al.: Learning multiple layers of features from tiny images (2009)
8. Li, T., Sahu, A.K., Zaheer, M., Sanjabi, M., Talwalkar, A., Smith, V.: Federated optimization in heterogeneous networks. arXiv preprint arXiv:1812.06127 (2018)
9. Lin, Y., Han, S., Mao, H., Wang, Y., Dally, W.J.: Deep gradient compression: Reducing the communication bandwidth for distributed training. arXiv preprint arXiv:1712.01887 (2017)
10. McMahan, B., Moore, E., Ramage, D., Hampson, S., y Arcas, B.A.: Communication-efficient learning of deep networks from decentralized data. In: Artificial Intelligence and Statistics. pp. 1273–1282. PMLR (2017)
11. Wang, H., Sreenivasan, K., Rajput, S., Vishwakarma, H., Agarwal, S., Sohn, J.y., Lee, K., Papailiopoulos, D.: Attack of the tails: Yes, you really can backdoor federated learning. arXiv preprint arXiv:2007.05084 (2020)
12. Wu, Y., He, K.: Group normalization (2018)
13. Zhu, L., Liu, Z., Han, S.: Deep leakage from gradients. In: Advances in Neural Information Processing Systems. pp. 14774–14784 (2019)
14. Zhuang, W., Wen, Y., Zhang, X., Gan, X., Yin, D., Zhou, D., Zhang, S., Yi, S.: Performance optimization of federated person re-identification via benchmark analysis. In: Proceedings of the 28th ACM International Conference on Multimedia. pp. 955–963 (2020)