FedTree: A Fast, Effective, and Secure Tree-based Federated Learning System

Qinbin Li[1] qinbin@comp.nus.edu.sg
Yanzheng Cai[2] caiyz18@mails.tsinghua.edu.cn
Yuxuan Han[1] e0509777@u.nus.edu
Ching Man Yung[1] e0575726@u.nus.edu
Tianyuan Fu[1] futianyuan@u.nus.edu
Bingsheng He[1] hebs@comp.nus.edu.sg
[1] *National University of Singapore*
[2] *Tsinghua University*

Abstract

Federated learning has been a popular approach to enable collaborative learning without exchanging the raw data. While tree-based models, especially gradient boosting decision trees (GBDTs), have been widely used in reality, tree-based federated learning has not been well exploited. This paper presents a tree-based federated learning system called *FedTree*, which is designed to be effective, efficient, and secure. FedTree supports horizontal and vertical federated training of GBDTs with optional privacy protection techniques such as homomorphic encryption and differential privacy. Documentation, examples, and more details about FedTree are available at `https://github.com/Xtra-Computing/FedTree`.

Keywords: Federated learning, gradient boosting decision trees

1. Introduction

Federated learning (FL)(Kairouz et al., 2019; Li et al., 2019) enables multiple parties to collaboratively learn a machine learning model without exchanging their local data. It has been widely used to enable distributed training without violating the data regulations. On the one hand, most existing studies on FL (McMahan et al., 2017; Li et al., 2021) are based on gradient descent, which cannot support the training of tree-based models whose parameters are non-differentiable. On the other hand, tree-based models (e.g., GBDTs) show superiority in efficiency and interpretability compared with neural networks, and have won many awards in machine learning competitions (Chen and Guestrin, 2016). Thus, a tree-based FL system is necessary for the machine learning community.

In this paper, we introduce FedTree, which is an efficient, effective, and secure tree-based FL system. FedTree supports horizontal FL (data of different parties have the same feature space but different sample spaces) and vertical FL (data of different parties have the same sample space but different feature spaces) of GBDTs. Optionally, techniques such as homomorphic encryption (HE) (Paillier, 1999) and differential privacy (Dwork, 2011) can be used to protect the communicated messages or the model. Moreover, the parallel training algorithm can well exploit multi-core CPUs. Our experiments show that FedTree can achieve almost the same accuracy as centralized training, and is much more efficient than the other systems.

2. Background on GBDTs

While the capacity of a single decision tree is low, the GBDT is an ensemble of decision trees to boost the model performance. The GBDT model has won many awards in machine learning and data mining competitions (Chen and Guestrin, 2016). It has been widely used in real-world applications (Richardson et al., 2007; Kim et al., 2009; Burges, 2010).

The training of the GBDT model is in a sequential manner. In each iteration, a new tree f_t is trained to fit the residual between the prediction and the target. Formally, given a loss function l and a dataset $\mathcal{D} = \{(\mathbf{x}_i, y_i)\}_{i=1}^n$, GBDT minimizes the following objective function at the t-th iteration.

$$\begin{aligned}\tilde{\mathcal{L}}^{(t)} &= \sum_i l(y_i, \hat{y}_i^{t-1} + f_t(\mathbf{x}_i)) + \Omega(f_t) \\ &\approx \sum_i [l(y_i, \hat{y}_i^{t-1}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t)\end{aligned} \tag{1}$$

where $\hat{y}_i^{t-1}$ is the current prediction value, $\Omega(\cdot)$ is a regularization term, $g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ and $h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$ are first and second order gradient statistics on the loss function. The decision tree is built from the root until reaching the restrictions such as the maximum depth. The internal nodes (i.e., split values) and leaf nodes (i.e., prediction values) are determined to minimize Equation (1).

3. Related Work

There are many popular GBDTs libraries (e.g., XGBoost (Chen and Guestrin, 2016), LightGBM (Ke et al., 2017), CatBoost (Dorogush et al., 2018), ThunderGBM (Wen et al., 2020)). While these libraries have shown superior performance in the centralized setting, these libraries do not support FL of GBDTs.

There have been some FL systems such as FATE (Liu et al., 2021), TensorFlow Federated (tff), PySyft (Ziller et al., 2021), PaddleFL (pad), and FedML (He et al., 2020). However, most systems are designed for federated deep learning and do not support federated GBDTs. Although FATE includes federated GBDTs, it is not specially designed for the federated training of trees and it is very slow as we will shown in the experiments. To the best of our knowledge, FedTree is the first tree-specialized FL system.

4. Overview and Design of FedTree

Figure 1 shows the architecture of FedTree. FedTree has five components to enable the easy usage and deployment of FedTree in real-world scenarios.

Interfaces FedTree supports two kinds of interfaces for ease of use: command-line interface (CLI) and Python interface. Users only need to input the parameters (e.g., number of parties, federated setting) to define the training scenario. Then, FedTree can be launched with a one-line command.

Environment FedTree supports standalone simulation on a single machine and distributed computing on multiple machines. When there is a single machine, it can simulate the federated

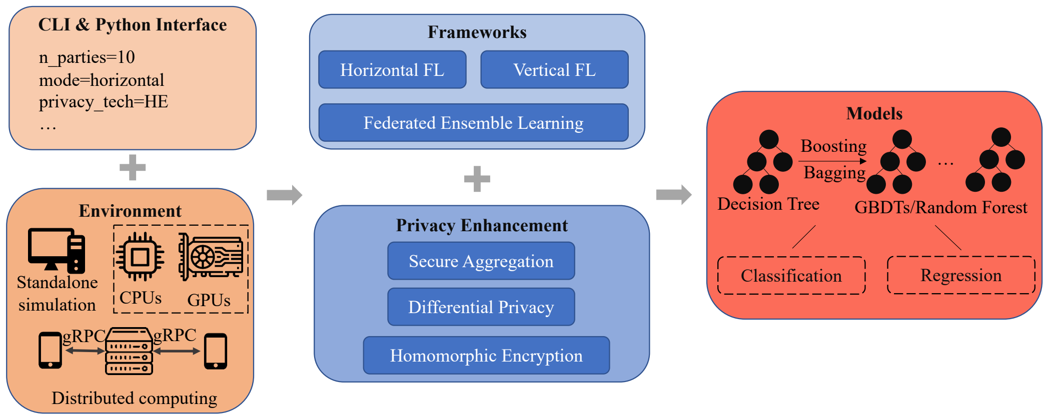

Figure 1: The architecture of FedTree

setting by partitioning a dataset into multiple subsets and treating each subset as the local data of a party. Where there are multiple machines for real federated setting, we adopt the party-server communication scheme and use gRPC[1] with a batched message passing design. Moreover, our system can utilize multi-core CPUs and GPUs to accelerate training through high-level parallelism.

Frameworks The core algorithms of FedTree are horizontal and vertical FL of GBDTs. In the horizontal FL setting, each party share the same feature space but different sample spaces. In the vertical FL setting, each party share the same sample space but different feature spaces. While the communication happens in every tree node in horizontal and vertical FL for model accuracy, we also design federated ensemble learning, which communicates in a per-tree manner to reduce the communication costs.

Privacy Enhancement The raw gradients are communicated in FL. To further enhance the privacy, we provide homomorphic encryption and secure aggregation to protect the communicated messages. Moreover, we provide differential privacy (Li et al., 2020) to protect the final model for releasing.

Models The training of a single tree is the building block of FedTree's models. With boosting (i.e., train each tree sequentially), FedTree can train GBDTs until reaching the given maximum number of trees. While GBDT is the key model that FedTree aims to support, FedTree also supports the training of random forests by training each tree independently. FedTree supports different loss functions for different tasks, such as cross-entropy loss for classification and square loss for regression.

5. Experiments

Due to page limit, we present preliminary experimental results to show the effectiveness and efficiency of our system. For more experiments, please refer to our document[2]. We use two UCI public datasets *adult* and *abalone* in our experiments. The number of trees is set to 50

1. `https://grpc.io/`
2. `https://fedtree.readthedocs.io/en/latest/index.html`

Table 1: Comparison of model performance between different systems.

datasets	XGBoost	ThunderGBM	FedTree-Hori	FedTree-Hori+SA	FedTree-Verti	FedTree-Verti+HE	SBT-Hori	SBT-Verti
a9a	0.914	0.914	0.914	0.914	0.914	0.914	0.912	0.914
abalone	1.53	1.57	1.57	1.57	1.56	1.57	1.56	1.56

Table 2: Comparison between the training time per tree (s) of FedTree and FATE. The speedup is the computed by the improvement of FedTree-Hori+SA over SBT-Hori and FedTree-Verti+HE over SBT-Verti.

datasets	FedTree-Hori	FedTree-Hori+SA	SBT-Hori	Speedup	FedTree-Verti	FedTree-Verti+HE	SBT-Verti	Speedup
a9a	0.09	0.098	8.76	89.4	0.11	5.25	34.02	6.48
abalone	0.11	0.19	7.7	40.5	0.05	7.43	15.7	2.11

and the depth of tree is set to 6. The learning rate is set to 0.1. The number of parties is set to 2 by default. We use "FedTree-Hori" to denote horizontal FedTree and "FedTree-Verti" to denote vertical FedTree. We use "HE" to denote homomorphic encryption and "SA" to denote secure aggregation. The experiments are conducted on a server with two Xeon E5-2640 v4 10 core CPUs.

5.1 Effectiveness

To show the effectiveness of FedTree-Hori and FedTree-Verti, we compare them with existing GBDT systems including XGBoost (Chen and Guestrin, 2016) and ThunderGBM (Wen et al., 2020). Moreover, we compare it with the federated GBDTs from FATE (Liu et al., 2021) denoted as SBT-Hori and SBT-Verti. As shown in Table 1, we report AUC for a9a and RMSE for abalone. We can observe that all systems have a very close model performance. FedTree-Hori and FedTree-Verti are loseless compared with the centralized GBDT training. Moreover, privacy techniques (i.e., SA and HE) do not affect the model performance while they provide protections on the exchanged messages during training.

5.2 Efficiency

To show the efficiency of FedTree, we compare FedTree with SBT from FATE. Table 2 reports the training time of the studied approaches. Note that FedTree-Hori+SA achieves the same security guarantees as SBT-Hori and FedTree-Verti+HE achieves the same privacy guarantees as SBT-Verti. We can observe that FedTree is much faster than SBT from FATE under the same privacy guarantees. The speedup is significant especially for horizontal FL. Moreover, FedTree provides flexible privacy configurations and the users can turn off the privacy techniques to achieve faster federated training while the local data are still not transferred.

6. Conclusion

In this paper, we present FedTree, which is an efficient, effective, and secure tree-based FL system. FedTree provides an easy-to-use platform for researchers and industries to conduct tree-based FL. We believe FedTree will make a significant contribution to the machine learning community.

Acknowledgement

We thank Wentao Han for providing computing resources.

References

Paddlefl. `https://github.com/PaddlePaddle/PaddleFL`. Accessed: 2022-06-06.

Tensorflow federated. `https://github.com/tensorflow/federated`. Accessed: 2022-06-06.

Christopher JC Burges. From ranknet to lambdarank to lambdamart: An overview. *Learning*, 11(23-581):81, 2010.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Anna Veronika Dorogush, Vasily Ershov, and Andrey Gulin. Catboost: gradient boosting with categorical features support. *arXiv preprint arXiv:1810.11363*, 2018.

Cynthia Dwork. Differential privacy. *Encyclopedia of Cryptography and Security*, pages 338–340, 2011.

Chaoyang He, Songze Li, Jinhyun So, Xiao Zeng, Mi Zhang, Hongyi Wang, Xiaoyang Wang, Praneeth Vepakomma, Abhishek Singh, Hang Qiu, et al. Fedml: A research library and benchmark for federated machine learning. *arXiv preprint arXiv:2007.13518*, 2020.

Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Keith Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *arXiv preprint arXiv:1912.04977*, 2019.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30:3146–3154, 2017.

Soo-Min Kim, Patrick Pantel, Lei Duan, and Scott Gaffney. Improving web page classification by label-propagation over click graphs. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 1077–1086. ACM, 2009.

Qinbin Li, Zeyi Wen, Zhaomin Wu, Sixu Hu, Naibo Wang, and Bingsheng He. A survey on federated learning systems: Vision, hype and reality for data privacy and protection. *arXiv preprint arXiv:1907.09693*, 2019.

Qinbin Li, Zhaomin Wu, Zeyi Wen, and Bingsheng He. Privacy-preserving gradient boosting decision trees. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 784–791, 2020.

Qinbin Li, Bingsheng He, and Dawn Song. Model-contrastive federated learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10713–10722, 2021.

Yang Liu, Tao Fan, Tianjian Chen, Qian Xu, and Qiang Yang. Fate: An industrial grade platform for collaborative learning with data protection. *Journal of Machine Learning Research*, 22(226):1–6, 2021.

H Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, et al. Communication-efficient learning of deep networks from decentralized data. In *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, 2017.

Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.

Matthew Richardson, Ewa Dominowska, and Robert Ragno. Predicting clicks: estimating the click-through rate for new ads. In *Proceedings of the 16th international conference on World Wide Web*, pages 521–530. ACM, 2007.

Zeyi Wen, Hanfeng Liu, Jiashuai Shi, Qinbin Li, Bingsheng He, and Jian Chen. ThunderGBM: Fast GBDTs and random forests on GPUs. *Journal of Machine Learning Research*, 21, 2020.

Alexander Ziller, Andrew Trask, Antonio Lopardo, Benjamin Szymkow, Bobby Wagner, Emma Bluemke, Jean-Mickael Nounahon, Jonathan Passerat-Palmbach, Kritika Prakash, Nick Rose, et al. Pysyft: A library for easy federated learning. In *Federated Learning Systems*, pages 111–139. Springer, 2021.