Onelearn: A Unified and Distributed Machine Learning Platform with High Performance

Yanshu Wang
Tsinghua University
wys17@mails.tsinghua.edu.cn

Tianfeng Liu
Tsinghua University
ltf17@mails.tsinghua.eud.cn

Shuai Wang
Tsinghua University
s-wang17@mails.tsinghua.edu.cn

Kaihui Gao
Tsinghua University
gkh18@mails.tsinghua.edu.cn

Liwei Jiang
Tsinghua University
jlw17@mails.tsinghua.edu.cn

Dan Li
Tsinghua University
tolidan@tsinghua.edu.cn

ABSTRACT

Software frameworks for machine learning play key roles in the design and development of machine learning applications. However, more and more complicated models are developed and data volume becomes very huge, therefor distributed training is necessary for machine learning applications. Moreover a machine learning framework usually supports a specific kind of machine learning algorithms, consequently a unified interface is required. In this paper, we introduce Onelearn, a Python-based, high-efficiency machine learning framework with model sharing, automatic resource management and unified interface. Onelearn provides a means of implementing a full range of distributed machine learning algorithms, including deep learning algorithms, traditional machine learning algorithms, etc.

1 INTRODUCTION

In recent years, machine learning has driven advances in many different fields and has become a core service in large companies[3]. Many machine learning frameworks have rise in recent years, such as Tensorflow[1], Scikit-learn[5], MXNet[2], Theano and Pytorch. But usually a single machine learning framework only supports a specific machine learning algorithm. This situation gives rise to high learning cost for machine learning developers. In addition, the model storage formats for different frameworks are incompatible, which makes it difficult to transfer model from different machine learning frameworks. Thus model sharing is significantly meaningful when it comes to cross-platform machine learning tasks.

The success of machine learning can be attributed to the design of more sophisticated machine learning models and the availability of Big Data. Sophisticated machine learning models have a large number of parameters. The scale of machine learning model and the volume of data are becoming increasingly large, which requires massive computations. The increment of computational complexity poses challenges to creating a distributed and high-performance machine learning system. In distributed machine learning, the parameters of machine learning models should be synchronized after some iterations of computation, which may cause heavy communication overhead. We can use efficient transport protocols such as RDMA(Remote Direct Memory Access) protocol to accelerate the parameter synchronization process. Besides, peer-to-peer communication framework can fully utilize network bandwidth and efficiently reduce network communication bottleneck compared to traditional parameter server architecture[4].

In distributed computing scenario, the resource management and the network configuration are intricate. However, existing machine learning frameworks only support allocating resource manually. For example, in TensorFlow, users have to specify IP addresses and cluster information in the training program, and users must start the program manually in each training worker which is really troublesome, especially when you have tens of thousands of nodes. Consequently it is desired to automatically manage resources and launch training tasks for machine learning system.

In this paper, we design and develop the Onelearn system. Compared with other machine learning systems, Onelearn provides a unified interface for a variety of machine learning algorithms and supports high-performance and peer-to-peer distributed training in heterogeneous clusters. Besides, supporting automatic resource management. It also provides model sharing interface, which enables Onelearn to share models with other frameworks.

2 DESIGN

2.1 Overall Architecture

As we mentioned earlier, Onelearn has four characters: unified programming interface, automatic resource management, distributed computation with highly efficient communication and model sharing with other frameworks. These four features are highly embodied in our system design. Figure 1 shows the architecture of Onelearn.

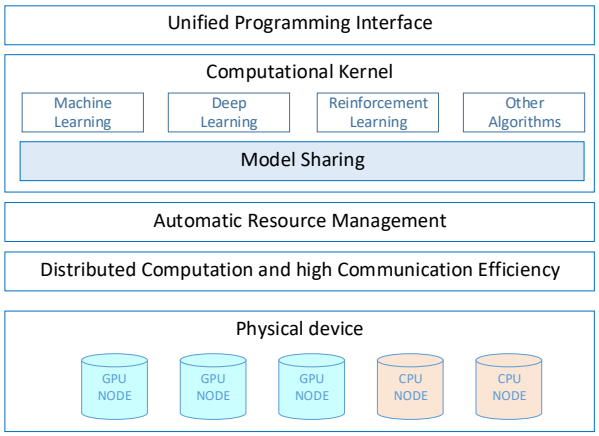

Figure 1: System architecture of Onelearn.

Onelearn system supports various algorithms and uses various computational kernels as backends. In Onelearn system, the physical resources are virtualized and can be automatically managed. Moreover, Onelearn will efficiently parallelize machine learning algorithms and use RoCE(RDMA over Converged Ethernet) protocol to accelerate the parameter synchronization.

2.2 Unified Programming Interface

Onelearn which utilizes existing best-practice computational kernels supports various machine learning algorithms including statistical machine learning algorithms, deep learning algorithms, reinforcement learning algorithms, etc. Onelearn will automatically choose the most appropriate computational kernels according to the algorithms. Onelearn uses Scikit-learn, Keras, etc. as computational kernels. These different computational kernels are targeted to corresponding machine learning algorithms.

Onelearn uses Scikit-learn computational kernel to implement machine learning algorithms. Our machine learning interface provides a large range of traditional machine learning algorithms i.e., classification, regression, clustering, dimensionality reduction, model selection and preprocessing.

Onelearn uses Keras as computational kernel for deep learning and reinforcement learning. Keras uses Theano, TensorFlow, and CNTK[6] as computational backends. Onelearn provides basic building blocks of neural network including neural network layers, optimization algorithms, and training tricks. Therefore developers have little burden to build models.

2.3 Automatic Resource Management

Automatic resource management in large-scale training jobs is a challenge. We design and implement automatic resource management module based on Kubernetes. Onelearn provides the ability to manage the resources transparently and automatically. The principles of the resource management module are as follow:

- Isolating resources, especially for GPU;
- Automatic addressing for the network of training cluster;
- Providing fault-tolerance capacity.

Training tasks are running in the containers, which guarantee isolation of resources. Pods which are a group of containers can be accessed by their corresponding services rather than IP addresses. Additionally, a distributed file system is mounted as persistent volumes(PV) to save training data, codes and training logs. Component Client which runs in a pod will allocate the resources transparently and robustly. Besides, Component Client maintains a certain amount of resources for each machine learning job and will reallocate resources when some workers break down, thus Resource reallocation can provide fault-tolerance capacity.

2.4 Distributed Computation and Highly-efficient Communication

Onelearn can automatically partition dataset and parallelize model. Users don't need to configure the distributed environment manually.

Distributed computing will result in a substantial communication overhead, thus an efficient transport protocol is important for communication. RoCE protocol, which shows significant help in improving the distributed computing performance[7], can also benefit Onelearn. We employ RoCE protocol to efficiently synchronize the parameters.

RoCE protocol is a kind of RDMA protocol characterized by kernel bypass and zero-copy which means the communication is processed by hardware and CPU don't need to be interrupted and the IO overhead is mitigated. Due to these features, Onelearn is able to achieve excellent performance in large-scale computing.

2.5 Model Sharing with Other Frameworks

Onelearn provides the ability to share models with other frameworks, to achieve that goal a unified model storage is required. Onelearn only supports CPU and GPU clusters, but in practice, we have diverse environments such as embedded systems and VR/AR devices, etc. Various frameworks have various optimizations on hardware environments, which benefit different machine learning applications. Therefore, we need to support various models on different frameworks to be compatible with different platforms.

We design unified model storage based on ONNX. ONNX provides a standard format for deep learning and machine learning models. We design and implement the frontend and backend of model sharing module in Onelearn. The frontend can covert Onelearn model to ONNX model, and backend can convert ONNX model to Onelearn model.

3 CONCLUSION

We have described Onelearn system and its architecture. Onelearn is a machine learning framework combining unified user interface with automatic resource management. Onelearn can efficiently and transparently run distributed machine learning algorithm. It is Python-based and supports various machine learning algorithms. Our initial experiences with Onelearn are encouraging. Some enterprises have put Onelearn into services and we have received inspiring feedback, while we continue to explore new design choices and more powerful architecture.

REFERENCES

[1] Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: a system for large-scale machine learning.. In *OSDI*, Vol. 16. 265–283.

[2] Tianqi Chen, Mu Li, Yutian Li, Min Lin, Naiyan Wang, Minjie Wang, Tianjun Xiao, Bing Xu, Chiyuan Zhang, and Zheng Zhang. 2015. MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. *CoRR* abs/1512.01274 (2015). arXiv:1512.01274 http://arxiv.org/abs/1512.01274

[3] Kim Hazelwood, Sarah Bird, David Brooks, Soumith Chintala, Utku Diril, Dmytro Dzhulgakov, Mohamed Fawzy, Bill Jia, Yangqing Jia, Aditya Kalro, et al. 2018. Applied Machine Learning at Facebook: A Datacenter Infrastructure Perspective. In *High Performance Computer Architecture (HPCA), 2018 IEEE International Symposium on*. IEEE, 620–629.

[4] Mu Li, David G Andersen, Jun Woo Park, Alexander J Smola, Amr Ahmed, Vanja Josifovski, James Long, Eugene J Shekita, and Bor-Yiing Su. 2014. Scaling Distributed Machine Learning with the Parameter Server.. In *OSDI*, Vol. 14. 583–598.

[5] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *Journal of machine learning research* 12, Oct (2011), 2825–2830.

[6] Frank Seide and Amit Agarwal. 2016. CNTK: Microsoft's open-source deep-learning toolkit. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2135–2135.

[7] Bairen Yi, Jiacheng Xia, Li Chen, and Kai Chen. 2017. Towards Zero Copy Dataflows using RDMA. In *Proceedings of the SIGCOMM Posters and Demos*. ACM, 28–30.