Distributed Computing System to Accelerate Deep Learning Training

1st Danqing Zhao
014546844
danqing.zhao@sjsu.edu

2nd Lin Cong
013748995
lin.cong@sjsu.edu

3rd Wei Lu
014336868
wei.lu@sjsu.edu

***Abstract*—Deep Learning has been the most popular technology in artificial intelligence. The power of computation has been stronger than ever, which contributes to the success of deep learning. However, even with more GPUs, it is still not enough for the more complex models, not to say a limited number of GPUs for a single computaion node. For the reasons stated above, the power of the distributed computing is considered to be the new energy.**

Previous researches has analysed the capability of the distributed computing adopted to train deep learning models [1]. Generally, there are two strategies, data parallelism and model parallelism [2]. Data parallelism means to distribute the data into different computation nodes around the network, while model parallelism tries to dismantle the model into parts. Both of these strategies has there advantages and disadvantages. They have been successfully adopted in the industry. Dean [3] had successfully developed a distributed system based on the model parallelism in google. However, all of these works are just a small step. It is stll worthy of researching on this area and developing more effective distributed systems.

To develop a more powerful distributed system, there are a lot of issues need to be solved. The first is the communication problem. Usually, most of the deep learning models has million parameters, resulting the model into very large size in a scale of GB. If the strategy needs the parameters to be shared between all nodes, the communication latency will be much significant. The second is about how to coordinate the global batch size and local batch size. According to the experiments of [1], if the local batch size is small enough, the communication latency will be beyond the latency of computing. But a large local batch size with small step size will harm the performance. Apart from the two major issues, we still need to consider the situation of one point failure and how to deal with fault tolerance. Fortunately, we have a lot of experience from the pioneers. Alexander [4] stole some ideas from the experience of HPC. Variants of Allreduce algorithm could be used to solve some of the issues.

The tools to train the deep learning model are easily accessed, for most of them are open sourced. The most popular frameworks are PyTorch and Tensorflow with their pros and cons. We can also use the other frameworks based on the framework that we will use for the network part. Based on these existing framework, we just need to design the interface with our distributed network, and decide what kind of data to communicate.

After we have talked about the deep learning configuration, we need to go through the network part. This part is much clear for our easy experiment environment. Our system will be developed in the local area network, which is built upon the personal router connnected with ethernet cable or wireless wifi. Due to the limitation of the resource, the network is heterogenous, which means each node has different computation power.Thus, there will be significant gap between the executation time of each node, resulting extra time to be wasted.To be specific, we will have to workstations in the network. Each has a different type of CPU, but both will be equiped with a scientific computation GPUs. How to fill the gap would be considered an advanced topic for our project. But it is not a big deal at the initial stage. It is obvious that our network will be based on the TCP/IP protocol. The details can be referenced to [6] and [7]. However, the exact model for the network is a big concern. We can adopt the start topology or ring into set up the network. Different topology has a rather different training and communication strategy. For example, if we would like to apply the Ring All Reduce,introduced by [5], we are more like to choose the ring topology.

The beneficial way for us to build this application should be from scratch, which could make us realize the details of how to implement such a complex system. However, due to the time limitation, we are considering use some tools to avoid dealing with some unnecessary operations, like service mesh. There are some mature implementations of service mesh, like consul, Istio, etc. One advantage is that most of the implmentations are open sourced, so we could have a deep look on them and make some flexible adaptions.

***Index Terms*—distributed computing, deep learning, data parallelism, model parallelism**

REFERENCES

[1] K. Janis, P. Franz-Josef, "Distributed Training of Deep Neural Networks: Theoretical and Practical Limits of Parallel Scalability"

[2] C. Karanbir,G.Manraj, D.Kuntal, "A Hitchhikers Guide On Distributed Training of Deep Neural Networks"

[3] D. Jeffrey, Corrado. Greg, etc "Large Scale Distributed Deep Networks"

[4] S.Alexander, B.Mike "Horovod: fast and easy distributed deep learning in TensorFlow"

[5] A. Gibiansky, "Bringing HPC techniques to deep learning"

[6] W.Richard Stevens, Bill Fenner, Andrew.M.Rudoff "UNIX Network Programming:The Sockets Networking API"

[7] E.Douglas "Internetworking with TCP/IP"