Scholar Publishing Group
Distributed Processing System
https://doi.org/10.38007/DPS.2020.010105
ISSN 2790-0916 Vol. 1, Issue 1: 34-41

Collaborative Optimization Method for Distributed Systems based on Deep Learning

Shaam Daparvar[*]

Chandigarh University, India

[*]*corresponding author*

Keywords: Deep Learning, Neural Network, Distributed System, Collaborative Optimization

Abstract: In the past decade, machine learning technology based on deep neural networks has made great progress, thanks to the continuous development of HPC hardware and software and practical applications. There are already a number of organizations and enterprises offering services to the public based on machine learning systems, such as face and speech recognition, photo optimization, and so on. Deep neural networks also require different computational power, so the demand for distributed neural network systems is also increasing. This paper mainly studies the collaborative optimization method of distributed systems based on deep learning. In this paper, a new distributed deep learning training system is designed and implemented by effectively combining cluster resource scheduling and distributed training based on the advantages of Pytorch in rapid neural network construction and parallel computing and cluster resource scheduling.

1. Introduction

Optimization theory is one of the important components in the theoretical research of cybernetics and operations research, and optimization problems widely exist in many engineering and scientific fields such as machine learning and artificial intelligence in real life [1-2]. Since the 1940s and 1950s, with the great progress of convex analysis theory and computer technology, optimization theory has also got a great development. With the rapid development of communication and microelectronics technology, emerged a large number of low-cost, high-performance miniature calculators and sensors, this for the realization of the distributed algorithm provides a good hardware basis, and distributed processing way more and more get the favour of people, make people turned to the study of distributed optimization algorithm (3-4). In simple terms, a distributed optimization is a complex or large optimization task disassemble into several smaller tasks, and put these small tasks assigned to a number of small processing system (i.e., a processing system), at the same time, these small processing system through communication and coordination between each other to do this complex or large optimization task. How to communicate and coordinate among these sub-processing systems is one of the key points and difficulties of distributed optimization algorithms [5]. The theory and application of distributed optimization have become an important research topic in the field of control and system science. How to design effective distributed optimization algorithms and analyze convergence and complexity has become one of the important tasks of optimization research.

Distributed optimization problems widely exist in many fields such as data regression, machine learning, model predictive control, smart grid, robot motion planning, state control system and so on [6]. For example, the problem of electric energy production and distribution in a power system can be formulated as a constrained distributed optimization problem, whose purpose is to find an optimal generation and transportation strategy that can minimize the total cost of power generation and transportation in the power system while meeting the capacity constraints of generation and transportation [7]. Many classical optimization algorithms (such as gradient descent method, conjugate gradient method, adjacent gradient method, etc.) can solve the above distributed optimization problems based on global information, but with the advent of the era of big data, these algorithms are increasingly unable to meet people's needs. This motivates scholars to design and study algorithms (such as distributed optimization algorithms) that can solve large-scale optimization problems [8].

Aiming at the hierarchical network architecture, this paper comprehensively studies the relevant neural network optimization technology, proposes the distributed neural network algorithm and architecture under the specific network architecture, tries to solve the problems encountered under the distributed operation in the actual scenario, and improves the performance of distributed deep neural network.

2. Optimization of Distributed Deep Learning System

2.1. Distributed Deep Learning

(1) Model parallelism and data parallelism

Distributed training can make full use of cluster resources to improve the speed. Generally, operations need to be carried out according to the matching of hardware resources and data scale, and computing tasks, sample division, distributed storage, distributed training and other aspects need to be considered [9]. Data parallel mode is mainly refers to the data set, work assigned to each node, different working nodes with multiple copies of a model, and then work nodes according to their respective local copies of data and model of distribution, the model for training, we call this kind of parallel mode "data parallel model", the classic data sample classification method has two kinds: The first method is based on "random sampling". The advantage of random sampling is to ensure that the local data on each machine is independent and identically distributed with the original data, but the disadvantages are as follows: 1. The training data is large and the computational complexity is high. 2. If the sampling times are too small, some samples may not be selected, resulting in the underutilization of training samples and affecting the accuracy rate. The second method is based on "scrambling and slicing". In this method, the training data is randomly scrambled, and then the scrambled data is divided into uniform small pieces according to the number of working nodes, and these small pieces of data are distributed to each working node. Compared with the random sampling method, although the distribution of data is slightly different from the original data distribution, the computational complexity of the scrambled segmentation method is much smaller than that of the global random sampling method. Moreover, the scrambled segmentation method can retain every sample and intuitively make more full use of samples. Data parallelism is a common way of distributed training, which has low requirements on configuration and can be achieved in multi-threaded environment [10-11].

As long as the model parallel mode refers to that the scale of the deep learning model is too large to be stored in the local memory, the model needs to be divided. Deploy different parts of the model separately into multiple nodes. For highly nonlinear neural networks, each worker node cannot complete the parameter training and update that it is responsible for relatively independently, and must rely on other worker nodes for training. When a node fails, the integrity of the model cannot be guaranteed, and model parallelism requires high configuration and costs too much. It is difficult for non-researchers to have such an environment [12-13]. Another is a combination of data parallelism and model parallelism called hybrid parallelism.

(2) Parameter server architecture

The parameter server architecture is a common architecture for distributed training. It has two roles: a parameter server (server) and a worker (worker). The parameter server is responsible for distributing the model and summarizing the gradient, and the worker is responsible for computing and maintaining communication with the parameter server. The updating model can be divided into synchronous updating and asynchronous updating [14].

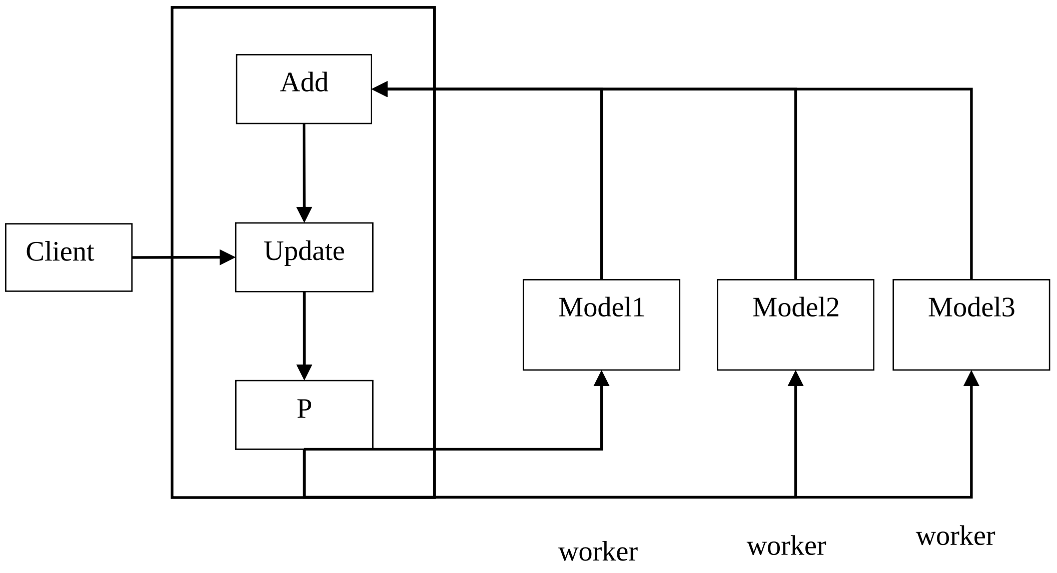

Figure 1. Synchronously update the architecture diagram

More shinyo synchronization in the iteration process, the work node first qualifying examination parameters obtained from the parameter server model, where each work node access is the same parameters, and then work nodes for training, after the completion of the training to the server sends parameters variation or gradient, parameters after the server to get all work node sends information to carry on the average, The global model on the parameter server is updated by gradient descent [15]. The synchronous update process is shown in Figure 1, and the formula is as follows:

$$g_{t+1} = \frac{1}{n}\sum_{k=1}^{k} g_t^k \tag{1}$$

$$w_{t+1} = w_t - \lambda g_{t+1} \tag{2}$$

Where, n represents the number of working nodes, K represents the number of nodes sent, λ is the learning rate, G represents the gradient, and wt+1 represents the T + L round model. The synchronization strategy has a clear structure and is not difficult to implement, so it is very suitable for small-scale clusters. However, with the increasing size of the cluster, problems continue to appear, such as configuration differences between machines, huge amount of network communication information, and the most fatal thing is that one node fails and the whole training will stop. Secondly, the computing and transmission delay is too long due to the imbalance of cluster resources.

To make up for the synchronous update time consuming and low reliability of shortcomings, puts forward the asynchronous update method (ASGD), parameters of server and work nodes using asynchronous communication way to update parameters, when finish work node training a batch when the server sends to the parameters, parameter server don't have to wait to finish all work node training, but rather an update again, This way the whole training won't stop even if a node fails.

2.2. Overall Design of Distributed System

As shown in Figure 2, the distributed neural network system in this paper is generally divided into two parts: the control side and the peer server. The main functions of the control end are to start the system, establish the network simulation topology, run the peer server process in the network simulation node, and control the actual operation of the peer server. The peer server is mainly responsible for receiving commands from the control side, performing actual calculations, communicating with other peer servers and other tasks. The whole system is mainly programmed on the control side, and the distributed neural network algorithm is constructed through the API exposed on the control side and combined with the neural network framework.

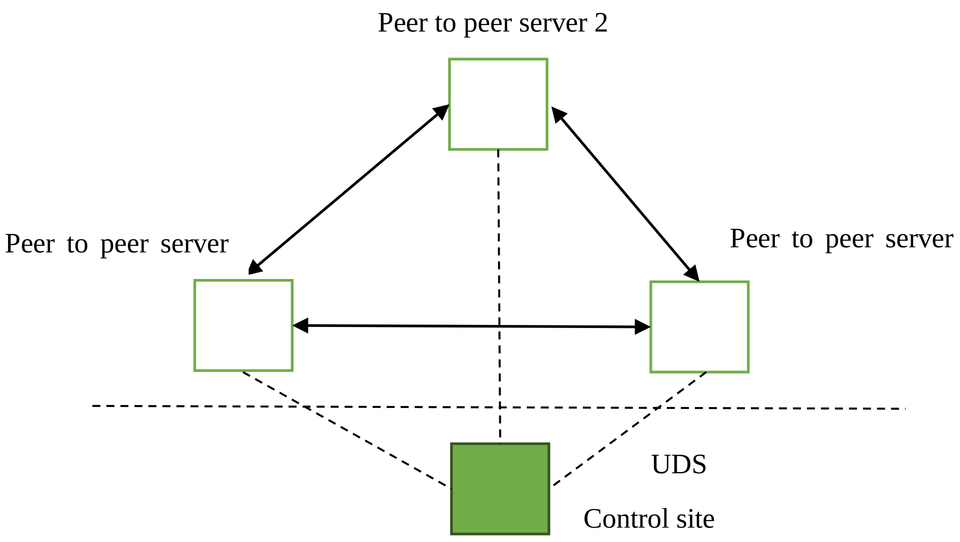

Figure 2. Overall system design

(1) Control end

The control side is the part of the brain that controls and directs in a distributed neural network system. By exposing the API, it programs users and completes most of the algorithm logic of the system [6]. When the system is running, the controller executes the algorithm logic written by the user and sends instructions to the peer server process in the corresponding network simulation node in real time to instruct it to calculate or communicate with other nodes.

There is only one centralized control process in the whole system, which runs outside the simulation network and communicates with the peer server through an independent delay-free communication channel. Therefore, the system can establish a simulation network with arbitrary topology structure, simulate a variety of network simulation scenarios, and ensure the ease of use of the system through a centralized control terminal. Because the control side is independent of the simulation network and does not actually belong to any part of the network topology, the centralized control side does not cause the so-called single point of failure.

(2) Peer server

A peer server is a program that runs in a network simulation node and is started by a controller. On the one hand, the peer server receives instructions from the control end through an independent channel without delay, performs calculations and other logical functions, and returns the system response (note, not the calculation results) [17]. On the other hand, the simulation network communicates with peer servers in other simulation nodes to transmit the calculation results.

The peer server does not contain specific algorithm logic, does not need user programming, and has full authority to receive the control of the control end, so the peer server is completely transparent to the user. Logically, the execution flow is similar to executing algorithms directly on network nodes. In terms of implementation, if the extra overhead of the system and the delay of the independent communication channel are small enough, the actual execution effect of the system is close to the direct execution of the algorithm on the network node. Therefore, the independent communication channel needs to remain delay-free (actually low latency).

Although there are different types of nodes such as terminal, edge and cloud in the hierarchical network structure, peer-to-peer servers are used uniformly in the distributed neural network system, and all nodes are completely peer-to-peer in the system logic [18]. By imposing different restrictions on each simulation node, such as computing power and delay, the nodes are made to imitate the behavior of each layer in the hierarchical network structure.

3. System Test Environment

The test environment of the system mainly includes hardware environment and related software environment. The underlying cluster of the system is mainly the Hadoop cluster environment. The hardware environment of the system is shown in Table 1:

Table 1. Hardware environment

Equipment	Quantity	Configuration
Master node	1	Intel Core i5-12600
		16G DDR4
		500G Hard disk
Work node	4	Intel Core i5-10400
		16G DDR4
		500G Hard disk

Software tools must be installed on each node in the cluster. The software environment is shown in Table 2:

Table 2. Software environment

Software	Version
Hadoop	2.0
Spark	2.0.1
Pytorch	0.4
Java	Jdk 1.7

4. Analysis of System Test Results

The system in this paper provides the function of setting work node optimizer and parameter server optimizer respectively, allowing users to set flexibly according to their needs. This experiment mainly verifies the effectiveness of this function by configuring different learning rates.

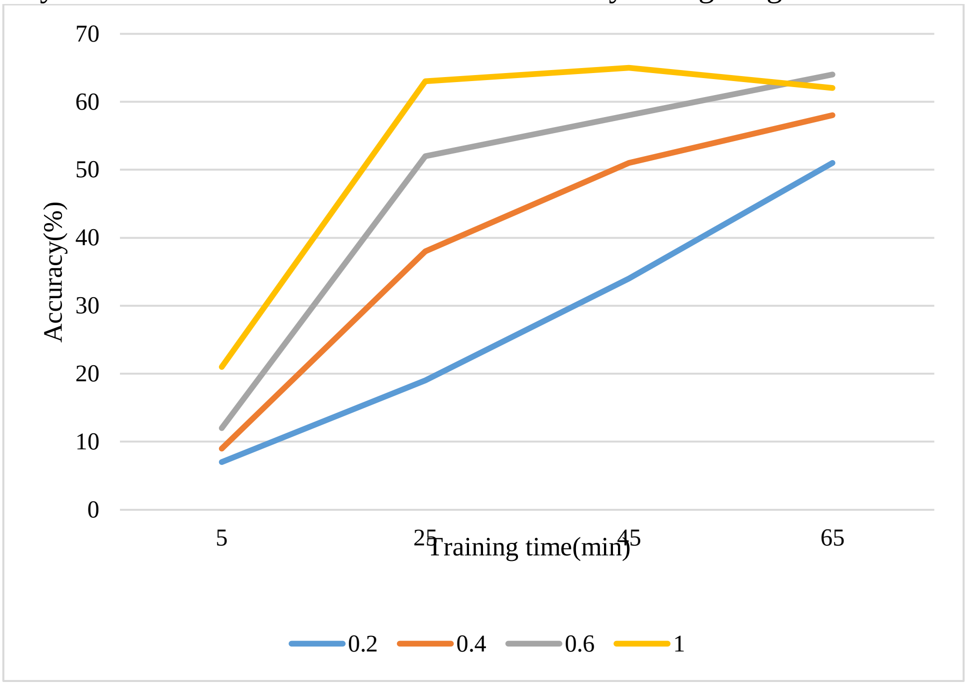

Figure 3. The training curve of different learning rate of parameter server

In the experiment, the optimization algorithm adopted by the working node and the parameter server node is stochastic gradient descent algorithm, and the parameter server node sets different learning rates to record the distributed training effect, as shown in Figure 3.

According to the figure, when the learning rate of the working node is fixed, setting the learning rate of the server with different parameters will also have different effects on the training effect. The small learning rate on the parameter server will compress the accumulated updates of the worker node, resulting in the waste of updates. If the learning rate of the parameter server is set to 1, it means that the update of the working node is fully utilized. In this case, although the convergence speed is fast, the update information from the working node may cause negative optimization due to the gradient obsolescence problem, resulting in unstable training process and large shock. Therefore, when the gradient obsolescence problem in distributed training is serious, the learning rate on parameter server can be appropriately reduced to ensure the stability of training. Users can flexibly select the optimizer algorithm and its parameter configuration to optimize the distributed training task.

5. Conclusion

Nowadays, neural networks have been widely used in speech recognition, computer vision and other fields. The complex practical requirements such as privacy protection and heterogeneous devices have prompted the research of distributed neural networks in federated learning. The layered network architecture derived from edge computing is a mature form of service application network. This paper focuses on the research of distributed neural network algorithms and systems for hierarchical network architecture, which is helpful to build practical applications and services related to distributed neural networks, and facilitate the integration of edge computing and artificial intelligence. Due to the limited ability of individuals, on the basis of the existing work, we need to make the following improvements and improvements: further optimize the distributed neural network simulation system, reduce its performance loss, improve the training speed, so as to reflect the algorithm and model performance more accurately.

Funding

This article is not supported by any foundation.

Data Availability

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Conflict of Interest

The author states that this article has no conflict of interest.

References

[1] Chagraoui H, Soula M. Multidisciplinary collaborative optimization based on relaxation method for solving complex problems:. Concurrent Engineering, 2020, 28(4):280-289. https://doi.org/10.1177/1063293X20958921

[2] Ines A, Joao X. DJAM: distributed Jacobi asynchronous method for learning personal models. IEEE Signal Processing Letters, 2018, PP(9):1-1.

[3] Mao W, Feng W, Liang X. A novel deep output kernel learning method for bearing fault structural diagnosis. Mechanical Systems and Signal Processing, 2018, 117(FEB.15):293-318. https://doi.org/10.1016/j.ymssp.2018.07.034

[4] Sattler F, Muller K R, Samek W. Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization Under Privacy Constraints. IEEE Transactions on Neural Networks and Learning Systems, 2020, PP(99):1-13.

[5] Rabhi S, Semchedine F, Mbarek N. An Improved Method for Distributed Localization in WSNs Based on Fruit Fly Optimization Algorithm. Automatic Control and Computer Sciences, 2020,

55(3):1-16. https://doi.org/10.3103/S0146411621030081
[6] Kavousi-Fard A, Dabbaghjamanesh M, Jin T, et al. An Evolutionary Deep Learning-Based Anomaly Detection Model for Securing Vehicles. IEEE Transactions on Intelligent Transportation Systems, 2020, PP(99):1-9.
[7] Saravanan G, Yuvaraj N. Cloud resource optimization based on poisson linear deep gradient learning for mobile cloud computing. Journal of Intelligent and Fuzzy Systems, 2020, 40(1):1-11. https://doi.org/10.3233/JIFS-200799
[8] Hamdia K M, Ghasemi H, Bazi Y, et al. A novel deep learning based method for the computational material design of flexoelectric nanostructures with topology optimization. Finite Elements in Analysis and Design, 2019, 165(Nov.):21-30.
[9] Ko H, Pack S. Distributed Device-to-Device Offloading System: Design and Performance Optimization. IEEE Transactions on Mobile Computing, 2020, PP(99):1-1.
[10] Samala R K, Kotapuri M R. Distributed Generation Allocation in Distribution System using Particle Swarm Optimization based Ant-Lion Optimization. International Journal of Control and Automation, 2020, 13(1):414-426.
[11] Tabasi M, Asgharian P. Optimal operation of energy storage units in distributed system using social spider optimization algorithm. AIMS Electronics and Electrical Engineering, 2019, 3(4):309-327.
[12] Sameti M, Haghighat F. Optimization of 4th generation distributed district heating system: Design and planning of combined heat and power. Renewable Energy, 2018, 130(JAN.):371-387. https://doi.org/10.1016/j.renene.2018.06.068
[13] Giri S, Mondal A K, Bera P. Design of PIDD controller for Hybrid Distributed Generation System using Social Spider Optimization Algorithm. Indian Science Cruiser, 2019, 33(3):27. https://doi.org/10.24906/isc/2019/v33/i3/185420
[14] Altsybeyev V, Kozynchenko V. Development of the distributed information system for the cooperative work under the design and optimizationof charged particle accelerators. Cybernetics and Physics, 2019(Volume 8, 2019, Number 4):195-198.
[15] Mughees M, Awan F G, Mughees A. Volt/VAr Optimization of Distribution System with Integrated Distributed Generation. Mehran University Research Journal of Engineering and Technology, 2017, 36(1):117-128. https://doi.org/10.22581/muet1982.1701.11
[16] Tabasi M, Asgharian P. Optimal operation of energy storage units in distributed system using social spider optimization algorithm. International Journal on Electrical Engineering and Informatics, 2019, 11(3):564-579. https://doi.org/10.15676/ijeei.2019.11.3.8
[17] Baran S, Nima S, Reza A, et al. Optimization of synchronized frequency and voltage control for a distributed generation system using the Black Widow Optimization algorithm. Clean Energy, 2020(1):1.
[18] Nawaz M A, Raheem A, Shakoor R, et al. feasibility and optimization of standalone pv-biogas hybrid distributed renewable system for rural electrification: a case study of a cholistan community. Mehran University Research Journal of Engineering and Technology, 2019, 38(2):453-462. https://doi.org/10.22581/muet1982.1902.19