Optimization Strategies for Large Model Training in Distributed Cloud Computing Environment

Dayi Wang and Zhe He
Naval Research Institute, Beijing 100161, China

Keywords: Training, Distributed, Cloud Computing, Large Models, Optimize, Tactics.

Abstract: In the context of large model processing, the environment of distributed cloud computing has been improved, but the complexity of platform processing has also increased. This paper analyzes the distributed cloud computing environment with the help of a large model and proposes an optimization strategy. Through in-depth analysis of large models, distributed environments, and data shutdown mechanisms, strategies such as efficient allocation of available resources, large model compression, and training optimization are proposed. After practical application, it can be seen that the above optimization methods can be used to carry out effective large model training and achieve good results. Specific data shows that after optimization, the training time is shortened by more than 50%, the GPU utilization rate is increased to 95%, the distributed environment is reduced by 50%, and the overall performance is significantly improved. The final conclusion shows that this strategy can greatly improve the training efficiency in the distributed cloud environment, give full play to the advantages of large models, and be suitable for a variety of application scenarios.

1 INTRODUCTION

Due to the rapid development of large model technology, the current data processing capacity is growing explosively, making traditional model training methods backward and no longer able to meet the needs of efficient processing. Some researchers have proposed that large models can be used to improve the training speed, but such solutions often face the problems of large model waste and large model bottleneck. Some researchers have proposed that asynchronous updates and local model aggregation strategies can be used to solve such problems. Although this method will achieve a certain improvement effect in the delay of large models, it still cannot effectively solve the problem of serious imbalance in resource utilization. The results show that the efficiency of the large model can be significantly improved based on the optimization of the parallel large model structure, but this kind of method cannot solve the problem of large distributed environment. In view of these challenges, this paper proposes that an intelligent optimization algorithm can be used to combine data parallelism and model parallelism, and improve the training efficiency of large models through dynamic resource large model association, model compression training, and storage optimization. The advantage of this method is that it can improve the utilization rate of large models, and the large model compression method can be used to reduce the bandwidth occupation. It is hoped that this method can provide a reliable scheme for large model training.

2 RELATED WORKS

2.1 Distributed Cloud Computing Theory

The distributed cloud computing theory is one of the basic theories of large model training, which involves task distribution and collaboration among multiple large model nodes (Aung, Dhelim, et al. 2024). In a distributed cloud platform, large model tasks are split into multiple subtasks, which are each assigned to different large model nodes for processing. The application goal of this theory is to improve the overall efficiency of the large model of the cloud platform and control the resource balance among the nodes of the large model (Bachhav, Kharat, et al. 2024). Parallel large models and training mechanisms

Wang, D. and He, Z.
Optimization Strategies for Large Model Training in Distributed Cloud Computing Environment.
DOI: 10.5220/0013536100004664

In *Proceedings of the 3rd International Conference on Futuristic Technology (INCOFT 2025) - Volume 1*, pages 97-102
ISBN: 978-989-758-763-4

in distributed cloud computing have become the focus of whether large-scale task processing can be realized (Balashov, Kuprikov, et al. 2024). However, traditional distributed cloud computing still needs to face the problems of large model latency and data consistency, especially in the process of model training and parameter update, which may still be the bottleneck of cloud platform performance (Du, and Wang, 2024).

2.2 Theory of Large Model Parallelization

The parallelization theory of large models provides an efficient large model training method for large models, which mainly includes data parallelization and model parallelism. In data parallelization, the training data is divided into multiple subsets, each of which is trained on a different large model node (Gautam, Batajoo, et al. 2024). This enables the cloud platform to process large models in parallel and improve training efficiency without affecting the model structure. The goal of model parallelization is to split the model itself into multiple parts (Jayanetti, Halgamuge, et al. 2024), and to make each part possible to execute the large model on different nodes. This strategy is suitable for scenarios with extremely large model parameters, but the distributed environment is large, and it is necessary to use a training and coordination mechanism to reduce the delay of large models between large models (Lee, Ryu, et al. 2024).

3 METHODS

3.1 Optimization of Task Allocation for Large Models

In a distributed cloud large model environment, the resource efficiency of large model training directly affects the training speed and cost. The training of large models requires the rational use of the large model of each node, so it is necessary to dynamically allocate large model tasks during the training process, so that the task resource balance can be distributed to the major model nodes (Santos, Ghita, et al. 2024). To this end, the corresponding large model tasks can be allocated according to the processing power of each node through the application of the task large model parallel server, so as to avoid the idle situation of other nodes under the overload of some nodes. See Eq. (1) for details.

$$Load_i = \frac{Tasks_i}{Capacity_i} \tag{1}$$

In this formula, $Load_i$ represents the computational ratio of the first i node, which is the ratio of the amount of tasks performed by the node to its large model capabilities. $Tasks_i$ Represents the amount of large model tasks assigned to the first i node. $Capacity_i$ Represents the i large model processing power of the first node, which is usually related to CPU/GPU performance. Based on dynamic large model association, the resources of each node can be efficiently utilized, which in turn improves the overall training speed of large models and reduces the waste of resources of large models.

In a distributed environment with limited resources, parallelization may not be able to realize the potential of all nodes. Therefore, it is necessary to combine data parallelism and model parallelism strategies to achieve more efficient resource use. Data parallelism is based on splitting the large model set into multiple small blocks and allocating them to different nodes for training, while model parallelism splits the model into multiple nodes for parallel large models to improve the efficiency of training. For this, see Eq. (2).

$$T = T_{data} + T_{model} \tag{2}$$

In this formula, the T total training time is described. T_{data} Represents the time when the data is processed in parallel. T_{model} Represents the time when the model is processed in parallel. Based on the combination of the two parallel modes, the training time can be significantly reduced, and the resource utilization can be further improved.

In a distributed cloud large model environment, a large number of temporary data storage and invocation are involved in the training process of large models, such as intermediate large model results and model parameters. Utilizing an efficient distributed cloud platform, such as HDFS, ensures fast data reads and writes and reduces storage costs. For this, see Eq. (3).

$$S_{cost} = \frac{D}{T_{access}} \cdot C_{storage} \tag{3}$$

In this formula, the S_{cost} cost of storage is described. D Represents the amount of data. T_{access} Represents the time of data access. $C_{storage}$

Represents the cost per unit of storage. Based on optimizing the storage read and write speed and reducing storage redundancy, the storage cost of large model training can be significantly reduced.

3.2 Improve the Reliability of Large Models

Designing an obvious data shutdown mechanism can improve the reliability of the large model training process. In this paper, based on checkpoint preservation, the intermediate state of the model is saved in a certain training stage, and if a node fails, the training can be restarted from the nearest checkpoint. For this, see Eq. (4).

$$T_{resume} = T_{total} - T_{checkpoint} \tag{4}$$

In this formula, represents the T_{resume} time it takes to train for cloud data calls from checkpoints. T_{total} Represents the total time for the entire training session. $T_{checkpoint}$ Represents the training progress when the checkpoint is saved. This method can significantly reduce the retraining time caused by node feature data and further improve the reliability of the training process.

In order to ensure the stability and reliability of large models, this paper will also apply the redundant large model strategy. Based on replicating key large model tasks across multiple nodes, it avoids identifying training interruptions caused by the feature data of a node. In addition, resource balancing dynamically adjusts the amount of tasks on each node, thereby preventing individual nodes from being overloaded.

Distributed cloud computing training involves a large number of large models among large models. Network jitter and bandwidth bottlenecks may cause large models to fail. In order to ensure the reliability of the large model, the asynchronous large model mechanism can be used to allow each node to be large at different times to reduce the waiting time.

3.3 Distributed Environmental Control Optimization

In a distributed cloud large model environment, the distributed environment is one of the main bottlenecks of large model training. To do this, this needs to be optimized.

In large model training, the training of the model requires a large amount of bandwidth. The application of the local model aggregation method can reduce the amount of data per training. Each node should be able to aggregate a part of the model locally, and then train with other nodes to reduce the frequency of large models and the distributed environment. For this, see Eq. (5).

$$G_{sync} = \frac{1}{k}\sum_{i=1}^{k} G_i \tag{5}$$

In Eq. (5), it G_{sync} represents the globally trained model. G_i A local model that represents i the first node large model. k Represents the number of nodes. Based on model aggregation and delayed training, the amount of data per large model can be reduced to reduce the distributed environment.

In distributed training, the hierarchical large model structure can also reduce the distributed environment across nodes to a certain extent. In the first layer, the local network between nodes will be used to train the data, and in the second layer, the data will be trained across nodes. This hierarchical large model can reduce the frequency of large models across nodes and reduce the bandwidth requirements of large models. See Eq. (6) for this.

$$T_{total} = T_{local} + T_{global} \tag{6}$$

In this formula, is the T_{total} total large model time. T_{local} is the time between local large models. T_{global} is the global large model time across nodes. Based on this hierarchical design, the number and overhead of large models across the network will be minimized.

In order to reduce the amount of data transferred, the model and model parameters can be compressed and quantized. Based on compressing 32-bit floating-point numbers to 16-bit or even 8-bit, the amount of data in large models can be significantly reduced. While compression may introduce some loss of precision, it is usually negligible for large-scale training. See Eq. (7) for this.

$$S_{compressed} = S_{original} \cdot C_{rate} \quad (7)$$

In this formula, $S_{compressed}$ the amount of compressed data is represented. $S_{original}$ Represents the amount of raw data. C_{rate} Represents the compression ratio. Based on compression and quantization, the amount of data for each large model can be effectively reduced, and the overall efficiency of training can be further improved.

4 RESULTS AND DISCUSSION

4.1 Background of the Case

The main research object of this case is a large e-commerce company A, which uses distributed servers for processing, the number of servers is Alibaba, or NetEase Cloud, the server processing is 1TB, and the real-time processing method mainly studies the optimization of cloud recommendation cloud platform by large model training. Company A processes the browsing, clicking, and purchasing data of millions of users every day, and the recommendation cloud platform needs to use these large models to achieve real-time updates and provide personalized recommendations for users. However, when the number of users and data volume of company A increased rapidly, the original model training efficiency was significantly reduced, and it could not meet the needs of high-frequency updates. In order to improve the training speed of large models, the company introduced the optimization strategy designed this time to improve the performance of the recommendation cloud platform. In order to make the large model show superior performance in practical applications.

Table 1: Comparison of the usage of large models

Compute load resources	Legacy cloud platform	Distributed cloud platform
Distributed cloud computing	70%	90%
GET large model	60%	95%
KIMI large model	85%	90%

The communication platform of the large model and the cloud data output results are shown in Figure 1.

As can be seen from Figure 1, the large model is mainly used for processing large data, and the processing power is higher than that of the cloud computing environment, and the processing process is more complex. Large models can handle more complex data and are more holistic. Therefore, the data trained by the large model can meet the requirements of cloud computing and achieve distributed processing and analysis.

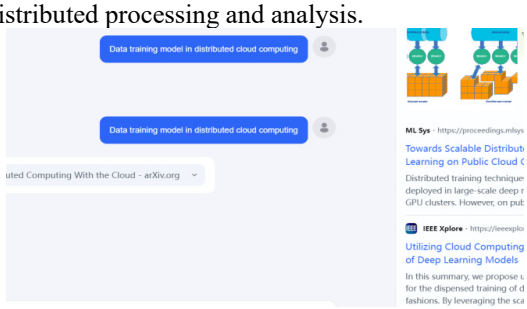

Figure 1: Technical analysis of KIMI and get large models

4.2 Optimization of Distributed Computing time for Large Model Training

Distributed cloud platforms offer significant advantages in terms of resource usage and time efficiency. From the perspective of large model utilization, the utilization rate of CPU and GPU under the distributed cloud platform has been significantly improved, especially the utilization rate of GPU has reached 95%, which indicates that the cloud platform can handle parallel large model tasks more efficiently. Moreover, the memory usage rate is also maintained at about 90%, indicating that under the architecture of distributed cloud computing, resource allocation is more reasonable. From the comparison of training time, it can be concluded that the training task that takes up to 30 hours to complete in the original cloud platform can be completed in only 10 hours in the distributed cloud platform. Table I shows that the utilization rate of large models has been greatly improved after the cloud platform upgrade, especially in the GPU usage of 95%.

Table 2: Comparison of time optimization in cloud computing environments

Model version	Before training on the cloud platform (min)	After training on the cloud platform (min)
GET large model	24±0.12	10±0.12
KIMI large model	30±0.32	12±0.72 AM
Other models	28±0.05 PM	9±0.15 AM

Table 2 shows that under the distributed cloud platform, the training time has been significantly reduced, with the fastest model version3 reduced from 28 hours to 9 hours. The time efficiency is improved by more than 50%, which is extremely critical for the training of large models that need to be updated frequently. The realization of this acceleration effect is mainly due to the enhancement of efficient large model association and parallel processing of resources, and the calculation process is shown in Figure 2。

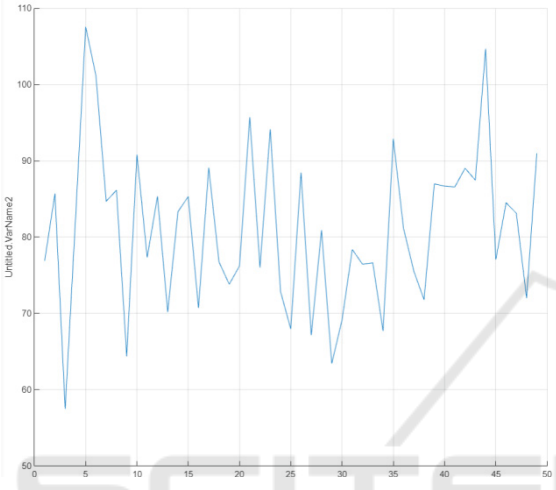

Figure 2: The training process of a large model

As can be seen from Figure 2, in the process of large model training, the storage node and the network large model module are integrated into a unified framework. This usually requires multiple aspects, such as large-scale model computing and the integration of distributed cloud platforms (Secrieru, Bogatencov, et al. 2024). Based on the seamless connection of these components, it can ensure resource balance, high-speed data access, and low-latency large model training among large model nodes. In addition, the cloud platform integration can further and dynamically adjust the resource allocation and training strategy of each module according to specific task computing and resource requirements. Based on large model data processing, cloud data tracking, and data association, the reliability of large model training will be greatly improved, and cloud data can be quickly called when feature data is identified (Verma, Taneja, et al. 2024). At the same time, the cloud platform integration also supports AI extension and large model association to cope with the balance of cloud resources in the training process.

4.3 Improvement of Distributed Computing Complexity by Large Model Loops

The reliability of large models is particularly critical in large-scale distributed training, because the feature data of any node has a greater possibility to affect the entire training process. In order to make the training stable and reliable, it is necessary to do more optimization on this. In a distributed environment, each large model node has a certain probability of identifying large model errors or interruptions due to hardware feature data, network problems, etc., so the calculation results are shown in Table 3.

Table 3: Simplified comparison of complexity of distributed computing environments

The complexity of the data	Before training on the cloud platform	After training on the cloud platform
Structured data	1±0.56TB	500±32.52GB
Qualitative data	200±15.63m/s	100±21.32m/s

Table 3 shows that the amount of data transfer is reduced based on large model compression, and the large model latency is reduced. In the analysis of the distributed environment, it can be found that the distributed cloud platform reduces the data transfer capacity from 1TB to 500GB by using compressed large models and optimized training strategies, and at the same time, the latency of large models is also reduced by 50%. This can greatly reduce the bandwidth usage and further improve the training efficiency between large models, as shown in Figure 3.

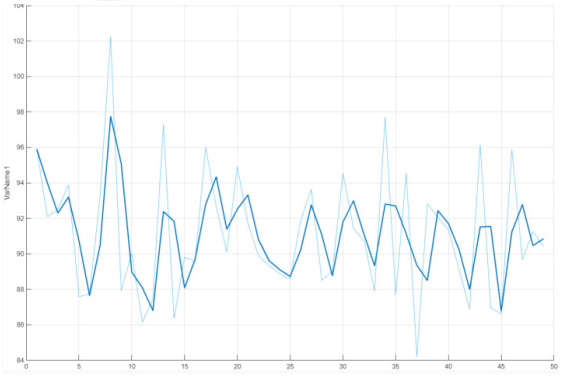

Figure 3: Comparison of the training of large models in the cloud computing environment

In Figure 3, the data fit degree of distributed computing is high, and the comprehensive judgment

of the data is realized through the processing of large models. It can be seen that the optimization strategy of large model training in the distributed cloud large model environment studied in this paper is very effective, which can ensure the good optimization effect of the distributed cloud platform in the large model, time efficiency and distributed environment, that is, the distributed cloud platform is particularly suitable for the large model training task.

5 CONCLUSIONS

This paper proposes an effective optimization strategy for large model training in a distributed cloud large model environment, which obviously solves the problems of high resource occupancy, difficult distributed computing and insufficient reliability in the traditional model training process. This strategy combines the efficient association, model compression, and training optimization of large models to achieve a comprehensive resource balance between distributed cloud computing and meet the requirements of fast data training, which greatly improves the speed of large model training and the overall performance of the cloud platform. In short, without using additional hardware resources, the stability and scalability of large models in a distributed environment can be ensured based on intelligent algorithms and optimization mechanisms. The research in this paper will provide a reliable and scalable solution for large model training, and it will be widely used in the field of artificial intelligence. Although the data in this paper is kept as large as possible, there are still some limitations, which can be expanded in the future.

REFERENCES

Aung, N., Dhelim, S., Chen, L. M., Ning, H. S., Atzori, L., & Kechadi, T. (2024). Edge-Enabled Metaverse: The Convergence of Metaverse and Mobile Edge Computing. Tsinghua Science and Technology, 29(3), 795-805.

Bachhav, A., Kharat, V., & Shelar, M. (2024). QOTUM: The Query Optimizer for Distributed Database in Cloud Environment. Tehnicki Glasnik-Technical Journal, 18(2), 172-177.

Balashov, N., Kuprikov, I., Kutovskiy, N., Makhalkin, A., Mazhitova, Y., Pelevanyuk, I., et al. (2024). Changes and Challenges at the JINR and Its Member States Cloud Infrastructures. Physics of Particles and Nuclei, 55(3), 366-370.

Du, L. Y., & Wang, Q. X. (2024). Metaheuristic Optimization for Dynamic Task Scheduling in Cloud Computing Environments. International Journal of Advanced Computer Science and Applications, 15(7), 590-597.

Gautam, B. P., Batajoo, A., & Shirator, N. (2024). A Proposal of JYAGUCHI Computing Platform to Realize ClouEdge (Cloud-Edge) and Serverless Architecture *. Journal of Information Science and Engineering, 40(1), 89-105.

Jayanetti, A., Halgamuge, S., & Buyya, R. (2024). Multi-Agent Deep Reinforcement Learning Framework for Renewable Energy-Aware Workflow Scheduling on Distributed Cloud Data Centers. Ieee Transactions on Parallel and Distributed Systems, 35(4), 604-615.

Lee, H., Ryu, J., & Won, D. (2024). Secure and Anonymous Authentication Scheme for Mobile Edge Computing Environments. Ieee Internet of Things Journal, 11(4), 5798-5815.

Santos, N., Ghita, B., & Masala, G. L. (2024). Medical Systems Data Security and Biometric Authentication in Public Cloud Servers. Ieee Transactions on Emerging Topics in Computing, 12(2), 572-582.

Secrieru, G., Bogatencov, P., & Degteariov, N. (2024). Extension of Distributed Computing Infrastructure and Services Portfolio for Research and Educational Activities. Physics of Particles and Nuclei, 55(3), 492-494.

Verma, R., Taneja, H., Singh, K. D., & Singh, P. D. (2024). Enhancing Data Analytics in Environmental Sensing Through Cloud IoT Integration. Journal of Climate Change, 10(2), 41-45.