International Journal of Internet, Broadcasting and Communication Vol.15 No.1 157-163 (2023)
http://dx.doi.org/10.7236/IJIBC.2023.15.1.157

IJIBC 23-1-19

Agent with Low-latency Overcoming Technique for Distributed Cluster-based Machine Learning

Seo-Yeon Gu*, Seok-Jae Moon**, Byung-Joon Park***

**Master, Department of Computer Science, Kwangwoon University, Korea.*
***Professor, Department of Artificial Intelligence Institute of Information Technology, KwangWoon University, Korea*
****Professor, Department of Computer Science, Kwangwoon University, Korea.*
E-mail: {gsy0207, msj80386, bjpark}@kw.ac.kr

Abstract

Recently, as businesses and data types become more complex and diverse, efficient data analysis using machine learning is required. However, since communication in the cloud environment is greatly affected by network latency, data analysis is not smooth if information delay occurs. In this paper, SPT (Safe Proper Time) was applied to the cluster-based machine learning data analysis agent proposed in previous studies to solve this delay problem. SPT is a method of remotely and directly accessing memory to a cluster that processes data between layers, effectively improving data transfer speed and ensuring timeliness and reliability of data transfer.

Keywords: *Cloud, Low-latency, Machine Learning, Unsupervised Learning, Data Analysis.*

1. INTRODUCTION

Communication between a server and a client in a cloud environment is performed by receiving information using various devices and protocols. Currently, companies are building data analysis systems according to business characteristics in a cloud environment [5, 6]. However, as business and data types become more complex and diverse, the demand for more efficient data analysis using machine learning is increasing [5, 6, 7]. In addition, communication in a cloud environment is greatly affected by network delay. Accordingly, when information received from a communication device is delayed, smooth data analysis is difficult [3]. In order to solve the delay, cloud environment service providers have introduced expensive equipment, TOE (TCP/IP Offload Engine) [1, 4], but this is also low in cost efficiency. In [3], the SPT (Safe Proper Time) method was proposed to solve the low latency relatively efficiently. SPT is a method of remotely and directly accessing memory to a cluster that processes data between layers, effectively improving data transfer speed and ensuring timeliness and reliability of data transfer.

[2] proposed a machine learning architecture that applied cluster-based reinforcement learning for big data

Manuscript Received: January. 8, 2023 / Revised: January. 14, 2023 / Accepted: January. 17, 2023
Corresponding Author: msj80386@kw.ac.kr
Tel:***-****-**** Fax: +82-10-916-4751
Professor, Department of Artificial Intelligence Institute of Information Technology, KwangWoon University, Seoul, Korea

analysis in a cloud environment. However, there was a limitation that data communication was not smooth after big data analysis was completed due to unstable network delay between the node cluster and the server. In this paper, the low-latency problem of data transmission is addressed by applying SPT [3] to the agent architecture proposed in [2].

The agent proposed in this paper consists of three layers: Machine Learning Processing Manager, Unsupervised Learning Manager, and Low-latency Management Layer. The Low-latency Management Layer consists of node *hardware/kernel handlers* and *sequential/batch/integration transmission modules*. The Low-latency Management Layer can solve the low-latency problem by applying a Safe Proper Time (SPT) transmission method.

The composition of this paper is as follows. Section 2 examines related studies, and Section 3 describes the function and performance algorithm of the Low-latency Management Layer. Section 4 describes agent performance analysis, and finally Section 5 concludes.

2. RELATED WORK

As the need for efficient data analysis in the cloud environment increases, [2] proposed an unsupervised learning-based cluster data analysis agent applied with reinforcement learning. This agent consists of two modules: Unsupervised Learning Manager and Machine Learning Processing Manager. Unsupervised Learning Manager creates a cloud environment by clustering nodes, and Machine Learning Processing Manager trains data sets transmitted from the server. [2] compared the execution time of the proposed agent system and the existing data batch processing system and measured the cases where the received data could not be received normally. As a result, in the former experiment, the proposed system performed analysis in less time than the batch processing system. In the latter experiment, the response failure rate gradually increased as the number of nodes constituting the agent increased, resulting in a delay in data transmission. This is because it takes a lot of time to adjust the proposed agent system in stages compared to the existing system. To this end, this paper attempts to overcome delay on the network by applying the SPT (Safe Proper Time [3]) transmission method as a way to stably manage the operating state of the network and resources.

3. MULTI AGENT OVERCOMING LOW-LATENCY BASED ON CLUSTERS IN CLOUD ENVIRONMENT

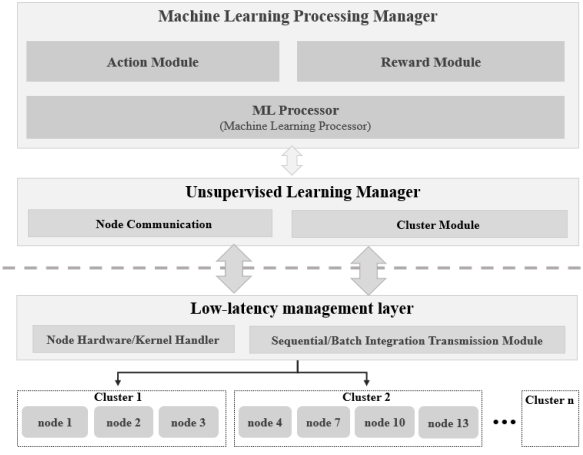

Figure 1. Proposal Agent Configuration

The agent consists of a Machine Learning Processing Manager, an Unsupervised Learning Manager, and a Low-latency Management Layer. Machine Learning Processing Manager consists of *Action Module, Reward Module, and ML_Processor*, and Unsupervised Learning Manager consists of *Node Communication* module *and Cluster Module*. The Low-latency Management layer consists of a *Sequential/Batch/Integration Transmission Module* and *Node Hardware/Kernel Handler*. Data transmission and analysis are performed through communication between the three managers in Figure 1. The following is a description of each module.

3.1 Machine Learning Processing Manager

The Machine Learning Processing Manager consists of *Reward Module* and *Action Module*, and is the main manager that controls the environment of the proposed system by judging the results of data analysis.

- *Reward Module*: *Reward* in a multi-agent system is the accuracy of the target query or analysis result. The system continues to run until the *Reward* is maximized.
- *Action Module*: *Action* in machine learning is an action to control the environment in a direction to maximize the *Reward*. Since the proposed agent analyzes the dataset based on the *Reward*, the *Reward Module* judges the *Reward* and transmits the data to the agent when further analysis of the data is needed so that the analysis is performed again.
- *ML_Processor* (Machine Learning Processor): *ML_Processor* sends a node communication request to the *Node Communication* module for cluster configuration.

3.2 Unsupervised Learning Manager

Unsupervised Learning Manager is a manager that configures a distributed cluster environment by managing communication between nodes.

- *Cluster Module*: *Cluster Module* configures a cluster by receiving node information from *Node Communication Module*. It continuously communicates with *Node Communication* module to create an optimal system environment.
- *Node Communication*: This module is responsible for communication between nodes. It checks the state of nodes on the network and checks which nodes are viable. It receives node information from *Cluster Module* and transmits data to the Low-latency Management Layer.
- *Data Collector*: *Data Collector* module collects the analyzed data from each agent, converts it into *Action*-type data, and transmits it to the *MainServer*.

3.3 Low-latency Management Layer

The low-latency management layer can solve the low-latency problem of the network by introducing the SPT method. The SPT method enables data to be transmitted without other operations such as kernel module loading. Depending on the size of each data set, the layer module selects the optimal transmission method to overcome communication delay among sequential transmission, batch transmission, and integration transmission.

- *Sequential/Batch Integration Transmission Module*: N data sets transmitted from the *Node Communication* module have different sizes. In this module, the size of each data set is measured so that an appropriate method can be selected among the three transmission methods.
- *Node Hardware/Kernel Handler*: This module fundamentally eliminates the copying of each layer of TCP/IP, which is one of the kernel sections. And to minimize the kernel section speed, the TOE (TCP/IP Offload Engine) module of the existing kernel is loaded and processed. For this reason, modules according to operating system versions and system types exist separately to solve risks and difficulties in maintenance.

3.4 Operation Mechanism

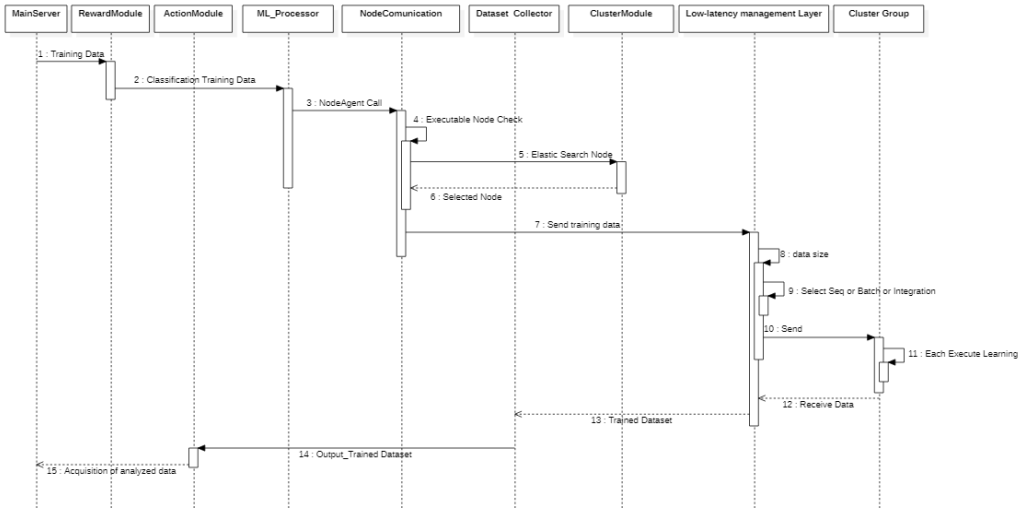

Figure 2. Sequence Diagram

Figure 2 shows the data and mechanism of the agent system of this paper as a sequence diagram. This mechanism is repeated until the analysis accuracy is highest, that is, until *Reward* is maximized.

1 - 6: Check the communication status of multiple nodes to create agents and node clusters. Then, the data received from the *MainServer* is divided into several(N) data sets.

7: *Send training data*: N data sets are sent to the Low-latency Management Layer.

8: data size: The size of each N data set received by the Low-Latency Management Layer is measured.

9 - 10: *Select Seq or Batch or Integration*: Select one of three data transmission methods based on the size of the data set measured in the *data size* step and transmit the data set to each agent.

11: Each Execute Learning: Perform analysis of the data set transmitted from the Low-Latency Management Layer.

12 - 15: This is the process of transmitting the data results analyzed in each cluster to the *MainServer*. Since the SPT method was introduced in the multi-agent system, remote direct memory access is possible, enabling faster data transmission. It is determined whether to repeat the *Action* sequence by determining the accuracy rate of the analyzed data.

3.5 Low-latency Management Layer Algorithm

Figure 3 shows an algorithm for optimal data transmission according to data size.

```
// Automatic selection algorithm for the most optimal method according to data size.
procedure start:
    duration_of_time : int_Variable;
    final_proc : str_Variable;
    proc : str_Variable;
    send : str_Variable; // Input data
begin
    if data_size <= N then
        proc_method begin: //Sequential processing
```

```
            data_len : int_Variable;
            seq_trans_module : str_variable;
            byte_send : str_variable;
            begin
                seq_trans_module :=write(fd.buff.nLen);
                while (data_len<=0) loop
                    byte_send :=send(fd, buffer,1);
                end loop;
            end seq;
        end
    else
        proc_method begin: //batch processing
            trans_receive : str_variable;
            trans_possible : bool_variable;
            trans_max_size : int_variable;
            batch_trans_module : str_variable;
            validation : str_variable;
          begin
              batch_trans_module :="write(fd,buffer,nLen)";
              if trans_possible='y' or else trans_possible='n' then
                  trans_max_size :=send(fd,buffer,nLen);
              else
              end if;
              validation :="Receivable status, line error status";
              if trans_receive='y'
                  trans_max_size :="send(fd,buffer,nLen)";
              else trans_receive='n'
             end if;
          end:
    end if;
    duration_of_time :="choice the best method";
    final_proc :="sequential/batch";
end;
```

Figure 3. Algorithm for selecting data transmission method according to data size

In this paper, the data transmission process is dealt with sequential, batch, and integration methods. In Figure 3, sequential transmission is a method of transmitting data from a module at once. However, the internal processing method has a structure in which processing is repeated according to the transmission data size, and is repeated until the data size is greater than 0. This increases CPU usage and affects the execution of other processes. This method is used when the data size is small. The batch transmission method assumes that the data transmission process is not repeated according to the data size. This is a method of transmitting data after checking the transmittable status, network status, and receivable status of the receiver through one module call. This minimizes CPU usage due to transmission because it transmits regardless of data size. Integration transmission is a method of selecting and processing the most suitable method for data. Depending on the hardware and data size, the broker chooses the most efficient method between sequential and batch transmission.

4. EXPERIMENTS AND RESULTS

In this paper, an experiment was conducted to measure the performance of the low-latency management layer of the proposed agent. As experimental data, floating population data provided by public institutions was used. A Recurrent Neural Network (RNN) model was used as a machine learning policy to analyze the dataset [8]. Depending on the size of the data in the low-latency management layer, the transmission method for each dataset is different. In this paper, in order to measure the performance, the time until the data set is received by each cluster was measured. Another experiment measured the response failure rate according to the number of nodes in the cluster. Response failure rate refers to the case where the result learned by *MainServer* is not normally reached. In this paper, we tried to prove the legitimacy of distinguishing the transmission method according to the data size by conducting a comparative experiment with agents of previous studies to which the low-latency management layer was not applied. The experimental environment is as follows.

- Experiment Environment: VMWare Ubuntu 22.04.1 LTS 5 nodes, Window Server 2019

Figure 4. Comparison of processing time with existing agent system

When the data size was small, the difference between the two agent systems was not large, but as the data size increased, the difference increased.

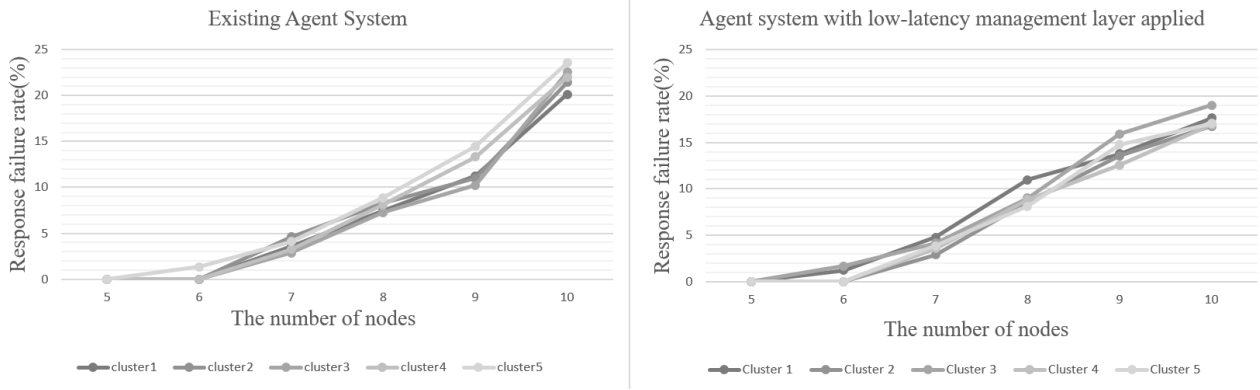

Figure 5. Comparison of response failure rate with existing agent system

Figure 5 is a chart comparing the response failure rate of the existing agent system [2] and the agent system proposed in this paper. It can be seen that the response failure rate of the agent of this paper is somewhat lower

than that of the existing agent system.

5. CONCLUSION

In this paper, we proposed a machine learning agent to overcome low latency in a cloud environment. Figures 4 and 5 show slightly improved performance compared to the agent system in previous studies by using real data to the agent system. The proposed agent solves the communication delay between the cluster and *MainServer* by distinguishing the data transmission method according to the size of the data. In addition, since the relatively efficient operation of resources is possible compared to the existing system, the response failure rate is also reduced. However, it is necessary to consider data transmission delay that may occur when data packet encryption is performed for security in a cloud environment. Research should be conducted to analyze problems caused by data encryption and to derive solutions.

ACKNOWLEDGMENT

This work is financially supported by Korea Ministry of Environment(MOE) Graduate School specialized in Integrated Pollution Prevention and Control Project.

REFERENCES

[1] Dong-Jae Kang, Chei-Yol Kim, Kang-Ho Kim and Sung-In Jung, "Design and implementation of kernel S /W for TCP/IP offload engine(TOE)," The 7th International Conference on Advanced Communication Tec hnology, 2005, ICACT 2005., 2005, pp. 706-709
DOI: https://doi.org/10.1109/ICACT.2005.245966.

[2] S.-Y. Gu, S.-J. Moon, and B.-J. Park, “Reinforcement learning multi-agent using unsupervised learning in a distributed cloud environment,” International Journal of Internet, Broadcasting and Communication, vol. 14, no. 2, pp. 192–198, May 2022.
DOI: https://doi.org/10.7236/IJIBC.2022.14.2.192

[3] Keun-Heui Kim, Seok-Jae Moon, Chang-Pyo Yoon, Dae-Sung Lee, “Study on Low-Latency overcome of Stock Trading system in Cloud”, Journal of the Korea Institute of Information and Communication Engine ering,18(11),2658-2663, 2014
DOI: https://doi.org/10.6109/jkiice.2014.18.11.2658

[4] J. S. Chase, A. J. Gallatin and K. G. Yocum, "End system optimizations for high-speed TCP," in IEEE C ommunications Magazine, vol. 39, no. 4, pp. 68-74, April 2001
DOI: https://doi.org/10.1109/35.917506.

[5] K. Al-Gumaei, A. Müller, J. N. Weskamp, C. S. Longo, F. Pethig and S. Windmann, "Scalable Analytics Platform for Machine Learning in Smart Production Systems", 2019 24th IEEE International Conference o nEmerging Technologies and Factory Automation (ETFA), 2019, pp. 1155-1162,
DOI: https://doi.org/10.1109/ETFA.2019.8869075

[6] Giuseppe Aceto, Valerio Persico, Antonio Pescapé, “Industry 4.0 and Health: Internet of Things, Big Data, and Cloud Computing for Healthcare 4.0 ”, Journal of Industrial Information Integration, Volume 18, 2020
DOI: https://doi.org/10.1016/j.jii.2020.100129.

[7] M. Amani *et al*., "Google Earth Engine Cloud Computing Platform for Remote Sensing Big Data Applicati ons: A Comprehensive Review," in *IEEE Journal of Selected Topics in Applied Earth Observations and R emote Sensing*, vol. 13, pp. 5326-5350, 2020
DOI: https://doi.org/10.1109/JSTARS.2020.3021052

[8] D.-Y. Shin, “Machine Learning Based Architecture and Urban Data Analysis - Construction of Floating P opulation Model Using Deep Learning -,” Journal of KIBIM, vol. 9, no. 1, pp. 22–31, Mar. 2019.
DOI: https://doi.org/10.13161/kibim.2019.9.1.022