Abstract: Along with the fourth industrial revolution, smart factories are receiving a great deal of attention. Large volumes of real-time data that are generated at high rates, especially in industries, are becoming increasingly important. Accordingly, the Industrial Internet of Things (IIoT), which connects, controls, and communicates with heterogeneous devices, is important to industrial sites and is now indispensable. To ensure the fairness and quality of the IIoT with limited network resources, the network connection of the IIoT needs to be constructed more intelligently. Many studies are being conducted on the efficient use of the resources that are imposed on IIoT devices. Therefore, in this paper, we propose a collaboration optimization method for heterogeneous devices that is based on cloud–fog–edge architecture. First, this paper proposes a knowledge distillation-based algorithm that can collaborate on cloud–fog–edge computing on the basis of distributed control. Second, to compensate for the shortcomings of knowledge distillation, we propose a framework for combining a soft-label-based alarm level. Finally, the method that is proposed in this paper was verified through several experiments, and it is shown that this method can effectively shorten the response time and solve the problems of existing IIoT networks, and that it can be efficiently applied to heterogeneous devices.

Keywords: IIoT; distributed computing; knowledge distillation; deep learning; smart factory; heterogeneous device

1. Introduction

The Internet of Things (IoT) refers to “things” that can be applied to various industries and service fields by sharing data with other objects, such as network-connected devices, wearable devices, mobile devices, smart home devices, and industrial equipment [1]. In other words, it refers to things that are connected and equipped with sensors, software, and other technologies that can send and receive data to and from other things. In addition, with the development of sensors, improvements in device performances, and network development, the IoT is a technology that is attracting attention as a paradigm that provides networking with many things, sensors, and smart things [2–4]. The IoT can collect various data sources through the network communication between devices in real time, and it requires a lot of computational capacity and memory to store a large amount of generated data. However, it is difficult to store and process large amounts of data on edge devices as they may have limited storage and computing power. Even if a wide range of data are collected, storing and using such big data can be extremely time consuming, and the increased energy consumption can shorten the battery life of edge devices.

To solve the above problem, research was conducted to centrally offload the data between devices that are collected based on applying IoT network communication to cloud computing [5,6]. A cloud computer is a computer that is connected to the cloud, and that is not a local computer, and cloud computing refers to a technology that provides computer resources (network, server, database, etc.), such as data storage space and computing power,
when necessary [7]. This can reduce the cost of managing multiple devices. In other words, users can use and store data easily because they only need to manage one cloud, and the advantage is that there is no need to build a server for data collection and processing. As a cloud system, the IoT enables traffic management, intelligent security, large-scale sensing, efficient data transmission, and data sharing and analysis. As such, the IoT can be applied in various fields.

However, bottlenecks may occur because of the increase in the traffic on the Internet and on networks, the development of communication technology, and the large amount of data that are generated from various objects. With the development of the IoT, various devices are now used, and they communicate with each other, and the IoT, which once played a small role, has become much more important and is widely applicable. When many IoT devices are used in a variety of ways, a bottleneck can easily occur [8]. Some IoT devices generate a small amount of data and require only a short response time, while others generate a large amount of data and require many responses. This overloads the network and can easily become a bottleneck. Cloud computing is not efficient enough to support these applications, and because many IoT devices are bundled into one cloud, it becomes easier to produce bottlenecks and to overload the network [9–11]. Moreover, if the cloud server is attacked or if the data on the server are damaged because of a disaster, the information that has not been backed up in advance cannot be restored.

To solve the above cloud centralization problem, cloud-, edge-, and fog-based networks have been studied in recent years. Muneeb et al. [12] propose a data analysis system improvement architecture that is based on a real-time IoT device for a multilayer fog computing platform to improve the existing IoT-based network system. Xiaolong et al. [13] propose Non-Dominated Sorting Genetic Algorithm III (NSAG-III) to solve multitarget optimization problems through the optimization of the execution time and the energy consumption of mobile devices in an IoT-enabled cloud–edge computing environment. Deyin et al. [14] propose HierTrain, which is a hierarchical edge AI framework that efficiently performs DNN training tasks in a MECC (mobile–edge–cloud computing) architecture in order to solve the time problem and the massive-use-of-resources problem that are caused by cloud central control. Van-Nam et al. [15] propose a hierarchical edge–cloud publish/subscribe broker model that uses a two-layer routing scheme to alleviate the problem of event notifications being sent in large-scale IoT systems with distributed IoT devices.

With the development of the fourth industrial revolution, artificial intelligence, and networks, the function of the IoT in the industry is becoming more important. “IIoT” is the Industrial Internet of Things, which is a subcategory of the IoT that focuses on technologies and the methods that are used intensively in industrial fields. With the development of technology, the performances of many of the devices that are used in industrial sites have improved, and, as the latest technologies (machine learning, 5G network, fog and edge computing) are applied, the scale of the IIoT is increasing.

Recently, various research works with regard to the IIoT are being conducted in accordance with these developments. Harmatos et al. [16] suggest a method for evaluating the system by identifying and analyzing the interworking aspects of 5G, TSN (time-sensitive networking), edge computing areas, and various communication and computing fields to implement smart manufacturing. D.W and W.C. [17] researched the minimization of multiple levels of interference, such as the energy consumption and the power control, by using an intelligent dynamic spectrum resource management optimization algorithm-based decision in order to handle the vast amount of sensing data of the IoT in the industry. To solve the existing problem, intelligent dynamic real-time spectrum resource management was implemented by applying data-mining case-based reasoning. Chengjia Wang et al. [18] propose a heterogeneous brainstorming (HBS) method that is based on a knowledge distillation mechanism in cyber physical systems (CPS) and cloud–edge frameworks for federated heterogeneous distillation.
This paper proposes an alarm-level-based double verification framework that is applicable to heterogeneous equipment in order to efficiently implement an IIoT network. The paper makes the following specific contributions:

1. To solve the bottleneck problem, we propose a cloud–fog–edge-distributed network that connects with heterogeneous devices in industrial sites that is based on a fog-edge, rather than a cloud-based, central control;
2. We propose a knowledge distillation-based algorithm to efficiently apply deep-learning-based algorithms that require a lot of computing resources for the IIoT system;
3. For detailed verification, we propose a soft-label-based alarm level to provide a smooth network connection and an accurate verification method for the algorithm.

The paper is organized as follows: Section 2 describes related work; Section 3 describes the proposed framework and a detailed alarm level; Section 4 describes the hardware and software settings, the datasets, and the evaluation indicators, before proceeding with the experiment; Section 5 describes the experimental results; and, finally, Section 6 discusses the conclusions and plans for future research.

2. Background and Related Work

2.1. Cloud–Fog–Edge Computing

Cloud computing is a generic term for anything that is related to the provision of hosted services over the Internet. In other words, it refers to the activity of sharing and executing computer power, storage, and system resources that can be scaled across external or internal cloud networks. In general, a cloud-based computing framework has large and scalable computing power, unlike a general computer. Cloud computing falls into three main categories or types: infrastructure as a service (IaaS); platform as a service (PaaS); and software as a service (SaaS) [19–22]. These cloud computers have various advantages. First, it is possible to reduce costs by using the clouds that are provided by other companies. In this way, if you pay a certain amount to use the cloud from the outside, there is the advantage that you do not need to build a local server and maintain it. The second advantage is the high accessibility. The advantage of building a business on the cloud is that users can access their services from anywhere in the world through a web browser, regardless of what device they are using. In their research on the use of the cloud platform, Mohd. Saifuzzaman et al. [23] propose a deep-learning-based streetlight illuminance control and monitoring platform for energy efficiency.

While the existing cloud computing method centrally manages data processing and computation in locations that are physically separated from the data center, edge computing provides a distributed and open architecture processing performance with a system that is physically located near the devices or data sources [24–26]. Alternatively, it performs the role of collecting and communicating data by performing computing at a nearby location. This has the advantage of significantly shortening the data processing time and reducing the Internet and network bandwidth usage by efficiently processing data that can be processed in the vicinity of the source. In other words, since the latency and bandwidth requirements are minimized, the bottleneck, which is a disadvantage of centralized processing, can be eliminated. It also enables real-time decision making. Edge computing has an advantage in terms of security, as it can make its own decisions at the edge. Edge computing is a computer approach that is different from the existing cloud, and it is close to a symbiotic relationship that complements each problem, rather than replacing either the local or cloud-based control. Edge computing is also known as “cloudlet” computing, which is a small-scale platform in the cloud environment.

Fog computing is a computing architecture that selectively analyzes and utilizes data that is generated in the field around the point of the data generation instead of sending it to a remote data center [27]. Fog computing has features such as on-demand services, broadband network access, and the fast elasticity of cloud computing, and, at the same time, has the following features [28,29]: The first characteristic is that the frequency of the
delay occurrence is low because it is located close to the edge. This bandwidth-improving feature is the first motivation behind the emergence of fog computing, and it aims to smoothly support services to numerous network edge nodes that are physically widely distributed. The second feature is that it is possible to support interaction and mobility through real-time processing. Fog computing also serves as a bridge between the cloud and the edge, with the big difference being that cloud computing utilizes a network of remote servers on the Internet, and edge computing utilizes terminal devices or servers. Fog computing utilizes a local area network (LAN) in a network architecture.

2.2. Knowledge Distillation

“Knowledge distillation” is a concept that first appeared in 2014 and was proposed by Geoffrey Hinton [30]. “Knowledge distillation” refers to a method of transferring knowledge from a large pretrained model (teacher model) to a small model to be used in practice (student model), in terms of the model distribution. In a deep learning model, the more parameters and the more computation, the better the feature extraction, and, accordingly, the performance of the model improves. However, since it uses a lot of memory and requires a lot of computing power, the efficiency is reduced in actual use cases. In other words, knowledge distillation is a method that is used to improve the performance of a small model by transferring the knowledge of a large model to a small model in the learning process, without changing the structure of the small model, so that the small model can perform as well as the large model. The architecture of the knowledge distillation model is shown in Figure 1.

![Figure 1. The knowledge distillation architecture.](image)

Knowledge distillation is a procedure for model compression in which the student model is trained to match the teacher model. In this process, the loss function is minimized and is transferred from the teacher model to the student model, and the goal is to match the softened teacher logits of the output with the actual label. The logits apply the Softmax temperature scaling function, which effectively smooths the probability distribution and reveals the relationship between the classes that are learned by the teacher model. In classification tasks, neural networks generally use a Softmax output layer to transform the output into a probability, whereas knowledge distillation uses a smooth refinement of the Softmax. The Softmax equation of knowledge distillation is as follows:

$$q_i = \frac{\exp(Z_i/T)}{\sum_j \exp(Z_j/T)}$$  \hspace{1cm} (1)

$Z_i$ is the predicted value of each class, and $T$ is the parameter that is introduced by the knowledge distillation. Higher values of $T$ produce a smoother probability distribution.
2.3. Industrial Alarm Level

Alarm systems are very important in industrial sites for efficient operation, including in most factories, chemical facilities, and power plants. An alarm system is a tool that detects near-miss errors that return to the normal operating range by measuring the abnormal operating range from the normal operating range on the basis of the process variables. A great deal of research has been performed on the alarm level in industrial fields.

Jiandong et al. [31] provide an overview of the industrial alarm system and suggest the main causes of alarm overload in order to solve the problem of the performance degradation of the existing industrial alarm system. Kourosh et al. [32] propose an alarm modeling method that uses graph theory to solve the problem of a poor alarm system in the process industry. Al-Kharaz et al. [33] discuss a semiconductor alarm system to solve problems in the existing semiconductor manufacturing process and suggest management and evaluation methods to improve the alarm system. Syeda Farjana Shetu et al. [34] investigated the infection path and detection method of botnets, which are one of the cybersecurity threats of the IoT in the industry, and they propose DNS-based mining as a solution.

3. Cloud–Fog–Edge Alarm System Using Knowledge Distillation

This section introduces the overall architecture of the proposed method, the knowledge distillation techniques for the IIoT, and soft-label-based alarms.

3.1. Cloud–Fog–Edge Alarm-Level-Based Heterogeneous Device Knowledge Distillation

Solving problems through edge and fog collaboration, rather than with the cloud central processing method, is a simple and efficient method to apply to heterogeneous equipment. It does not infringe on the privacy of other devices and it can meet the requirements of edge-based distributed control. The overall architecture of the cloud–fog–edge alarm-level-based heterogeneous device knowledge distillation for the IIoT that is proposed in this paper, which is based on the abovementioned method, is shown in Figure 2.

Figure 2. The architecture of the cloud–fog–edge alarm-level-based heterogeneous device knowledge distillation for the IIoT.

According to the type and characteristics of the data that were collected from the equipment, the classification is carried out in the edge and the fog, and through this, data are collected in the cloud. The cloud composes the teacher model on the basis of the
collected data. Each model that is distilled through the teacher model is added to the fog as a student model. For example, if there is equipment from which image data is extracted in the field, the classification (image and time-series) is performed on the basis of the data that is collected at the edge. After that, it is delivered from the fog to the cloud, an image processing algorithm can be created and trained in the cloud, and the distilled model can be applied to the fog by distilling it. We distributed the network by collecting and classifying data using the edge and the fog. We also introduced teacher and student models for the fog and the cloud, respectively. This method can solve the asymmetry problem of a single network. In addition, the response time between each network can be shortened through the proposed distributed network. The distillation model that is created in the fog (i.e., the student model) is used in conjunction with the edge. An alarm level is applied on the basis of the results of the student model, and a secondary verification is performed in the cloud according to the alarm level.

Information on the interaction and the collaborative relationship between the cloud, the fog, and the edge is illustrated in Figure 3.

If there is no model in the cloud nor in the fog, as the data collection initiates, the data is stored in the DB of the cloud, and, when a sufficient amount of data is collected to make a model, a teacher model is created in the cloud. Since the cloud has a lot of scalable computing power, we can train a variety of models with more parameters to produce a teacher model. In this study, several candidate models were specified for the

Figure 3. Information flow chart on interaction and collaboration relationship.
utilization of the data that is frequently used in the industry and to solve problems. First, when time-series data, such as vibration and power spectra, need to be utilized, the long short-term memory (LSTM) and gated recurrent unit (GRU) models, which have commonly been used in many studies, were selected. Second, we selected CNN and autoencoder (AE) models when we utilized data from images and videos. After creating the teacher model, it goes through a distillation process and distributes the student model to the fog. Through the abovementioned process, a model, in accordance with the machine and the situation, is created, and several distilled models are stored in the fog.

After the above process is performed, the data collected from the edge is used by selecting an appropriate algorithm (one of the student algorithms that is produced in the fog), according to the data type. The level is set by a preset soft-label-based alarm that is constructed by using the output label from the model. At this time, if the level is 1, it is recognized as a safe state, the verification in the fog is finished, and the result is transmitted to the cloud DB. If the level is 2 or 3, the data is sent back to the cloud with a warning and in a dangerous state, and then a reverification is performed through the cloud teacher model.

3.2. Soft-Label-Based Alarm Level

The alarm level that is proposed in this paper was set on the basis of the soft-label results of the pretrained student model by using the MNIST dataset. After obtaining the soft-label mean and standard deviation of the student model output, the quantile method was applied. If you look for the distribution of the actual output value, it does not come out as a normal distribution because the data are asymmetric. After making the distribution as close as possible to a normal distribution model by using the natural logarithm of the histogram values, the warning level is set by using quantiles. The soft-label output after the knowledge distillation that is described above is a value between 0.00 and 1.00, and the value of each label that is predicted by the model is its output. At this time, a value close to 1 is the result value that is predicted by the model, and, in this paper, the alarm level is set as the range of the quantile on the basis of the maximum value close to 1. The ranges of each quantile are shown in Table 1.

| Quantile | Range |
|----------|-------|
| Q1       | 0.705 |
| Q2       | 0.853 |
| Q3       | 0.901 |
| Q4       | 0.954 |

The alarm level that is set on the basis of the above range is as follows: The maximum value of the soft label that is output (that is, the value that is predicted by the model), is matched to the above quantile range, and the alarm level is set is as follows:

\[
\begin{align*}
\text{level1} &= \max[\exp(Z_i/T)] \geq 0.95 \\
\text{level2} &= 0.85 \leq \max[\exp(Z_i/T)] < 0.95 \\
\text{level3} &= \max[\exp(Z_i/T)] < 0.85
\end{align*}
\]

(2)

Alarm Level1 (safe) is set for the range of values in Q4, and Alarm Level2 (warning) applies to the Q2 and Q3 quantiles that range between 0.704 and 0.954. Alarm Level3 (danger) is triggered by a range value less than Q2.

4. Experimental Environment

The methods and algorithms that are used in the proposed architecture were evaluated against various tasks, models, and datasets for effectiveness and validation. The specifications of the cloud, the fog, and the edge that were used for this experiment are presented in Table 2.
Table 2. The experiment environments.

|                        | Hardware Environment                  | Software Environment     |
|------------------------|--------------------------------------|--------------------------|
| **Cloud Computing**    | CPU: Intel Core i7-8700k, 3.7 Ghz,   | Windows                  |
|                        | six-core twelve threads 16 GB        | Tensorflow 2.0           |
|                        | GPU: Geforce RTX 2070                | Python 3.7               |
| **Fog Computing**      | CPU: Intel Core i7-8700k, 3.7 Ghz,   | Windows                  |
|                        | six-core twelve threads 16 GB        | Tensorflow 2.0           |
|                        |                                      | Python 3.7               |
| **Edge Computing**     | CPU: four-core ARM A57 1.43 GHz      | Linux                    |
|                        | 128 Core Maxwell 482 GFLOPs (FP16)   | Tensorflow 2.0           |
|                        |                                      | Python 3.6               |

4.1. Dataset

In order to test whether the proposed method is applicable to heterogeneous equipment, two datasets are used in this experiment. For the time-series data, the CWRU (Case Western Reserve University) bearing dataset, which is often used in experiments, is used. The CWRU data consisted of data recording steady-state operations and 10 faults (the inner raceway, rolling element, and outer raceway), which were measured at speeds of 1797–1720 RPM on a 0–3 horsepower motor, and which ranged from a 0.007-inch diameter to a 0.021-inch diameter. A total of 12,000 datasets were used in the experiment. Figure 4 shows the plots of the CWRU data.

The image data used the photographed images of the submersible pump impellers among the cast products. The data consist of two types: steady-state and defective-state data, and a total of 5000 image datasets were used in the experiment. Figure 5 shows the representative images that were used in the casting image dataset.

![Figure 4. Bearing vibration dataset time-series images.](image-url)

![Figure 5. Casting image dataset: (A) normal casting image; (B) abnormal casting image.](image-url)
In order to test whether the proposed method is applicable to heterogeneous equipment, two datasets are used in this experiment. For the time-series data, the CWRU (Case Western Reserve University) bearing dataset, which is often used in experiments, is used. The CWRU data consisted of data recording steady-state operations and 10 faults (the inner raceway, rolling element, and outer raceway), which were measured at speeds of 1797–1720 RPM on a 0–3 horsepower motor, and which ranged from a 0.007-inch diameter to a 0.021-inch diameter. A total of 12,000 datasets were used in the experiment. Figure 4 shows the plots of the CWRU data.

Figure 4. Bearing vibration dataset time-series images.

The image data used the photographed images of the submersible pump impellers among the cast products. The data consist of two types: steady-state and defective-state data, and a total of 5000 image datasets were used in the experiment. Figure 5 shows the representative images that were used in the casting image dataset.

Figure 5. Casting image dataset: (A) normal casting image; (B) abnormal casting image.

4.2. Evaluation Metrics

For the evaluation, the accuracy, the F1-Score, and the Matthews correlation coefficient (MCC) were used, which are based on true positive (TP), false positive (FP), false negative (FN), and true negative (TN), and which define the relationship between the model result and the actual result for the model evaluation MCC.

Accuracy is the most intuitive indicator. The problem, however, is that the performance can be skewed if the labels in the data are unbalanced. The formula for the accuracy is expressed as:

$$\text{Accuracy} = \frac{|TP| + |TN|}{|TP| + |FP| + |FN| + |TN|}$$  \hspace{1cm} (3)

The F1-Score is called the “harmonic mean”, and if the data labels are unbalanced, it can accurately assess the performance of the model. The resulting equation is given as follows:

$$F1 \text{ score} = 2 \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$  \hspace{1cm} (4)

The Matthews correlation coefficient (MCC) is the most widely used evaluation index for measuring the quality of multiclass classification models. It is also a well-rounded measure that can be used even if the classes are very different sizes. The equation for the MCC is:

$$\text{MCC} = \frac{|TP| \times |TN| - |FP| \times |FN|}{\sqrt{(|TP| + |FP|)(|TP| + |FN|)(|TN| + |FP|)(|TN| + |FN|)}}$$  \hspace{1cm} (5)

5. Experiment and Result

The experiment was conducted in a 30-epoch environment and by using the two previously selected datasets. It was divided into two stages, and the accuracies, the F1-Scores, and the MCCs of the algorithms that were used in the teacher model and the student model were measured and compared.

5.1. CWRU Dataset

To process the CWRU time-series dataset, the LSTM and GRU models, which are frequently used algorithms for time-series processing, were used. In order to match the conditions of the teacher model and the student model of the LSTM and GRU algorithms equally, they were designed using the same parameters. The data were input into the model as one-dimensional 400-length data. The input data passes through five layers, and the
overfitting problem is solved by arranging dropout layers for each layer. The final output for the teacher model is classified through the existing Softmax layer, and, for the student model, a knowledge distillation Softmax that is measured as the optimal temperature $T$ is used. Figure 6 shows the accuracy (ACC) and loss when the LSTM and GRU are used for the teacher and student models.

![Figure 6](image)

Figure 6. Accuracy and loss graphs of LSTM and GRU models for teacher and student models: (A) teacher LSTM and GRU ACC; (B) teacher LSTM and GRU loss; (C) student LSTM and GRU ACC; (D) student LSTM and GRU loss.

As is shown in Figure 6, the accuracy of the algorithm decreases after distillation. However, it shows reliable accuracy of over 90%.

Table 3 shows the results for the ACC, the F1-Score, and MCC of the Teacher and Student models.

| Model  | ACC Top1 | F1-Score | MCC    |
|--------|----------|----------|--------|
| Teacher| LSTM     | 97.81%   | 96.32% | 95.87% |
|        | GRU      | 95.52%   | 93.72% | 93.22% |
| Student| LSTM     | 96.77%   | 95.21% | 94.76% |
|        | GRU      | 92.97%   | 91.37% | 90.87% |

After distillation, the F1-Score and the MCC appear to decrease, but there is no significant difference, and the LSTM model shows higher accuracy.

Figure 7 shows the confusion metrics for the student model that was used in the fog. The higher the accuracy, the higher the probability of an accurate classification. However, in the case of the student model, the accuracy is lower than that of the teacher model, which makes it relatively difficult to classify the defects. In view of this, it is judged that
a secondary verification through the teacher model in the cloud is necessary for more accurate classification. On the basis of the preset alarm level, when Levels 2 and 3 occur in the fog, a signal is transmitted to the cloud for secondary verification. In the cloud, on the basis of this information, the same algorithm that was used by the student model in the fog is chosen as the teacher model in the cloud. After that, the second verification is carried out through the teacher model.

In order to confirm the secondary verification of the cloud, an analysis was performed according to the alarm level. Table 4 shows the results of the classifications on the basis of the results that were obtained by using 100 samples of test data that were randomly extracted for each model.

Table 4. Alarm-level test result percentages for the CWRU dataset.

| Model            | Level 1 (%) | Level 2 (%) | Level 3 (%) |
|------------------|-------------|-------------|-------------|
| Student (Fog)    | 90          | 4           | 6           |
| LSTM             |             |             |             |
| Teacher (Cloud)  | 96          | 2           | 2           |
| LSTM             |             |             |             |
| Student (Fog)    | 84          | 9           | 7           |
| GRU              |             |             |             |
| Teacher (Cloud)  | 92          | 5           | 3           |
| GRU              |             |             |             |

As is shown in the table, the student model in the fog returns relatively high percentages of Level2 and Level3 alarms. However, when the Level2 and Level3 preliminary alarms were secondarily verified with the teacher model in the cloud, the more accurate cloud models returned lower percentages of Level2 and Level3 alarms. By using the teacher models to perform a detailed inspection in the cloud, it was possible to reduce the final alarms by about 25%, relative to the fog alarms, while reducing the traffic and the cloud processing overheads.
5.2. Casting Dataset

For the casting dataset, we used CNN and AE models, which are algorithms that are often used for image processing as image datasets. Data are entered as $300 \times 300$. The CNN model uses five 2d convolution layers, two dropout layers, and two pooling layers. The final output is classified through the existing Softmax layer for the teacher model, and through a knowledge distillation Softmax that is measured as the optimal temperature 3 for the student model. The AE model plays the role of data augmentation, and the AE consists of three encoders and three decoders. The output value is added to the existing data and the input is added to the fully connected layer. At this time, the last layer is composed of the Softmax. In the final output, the teacher model uses the default Softmax, and the student model uses the Softmax temperature 3 after the knowledge distillation. Figure 8 shows the ACC and the loss when the CNN and AE models are used for the teacher and student models.

Figure 8. Accuracy and loss Graphs of CNN and AE models for teacher and student models: (A) teacher CNN and AE ACC; (B) teacher CNN and AE loss; (C) student CNN and AE ACC; (D) student CNN and AE loss.

As is shown in Figure 8, the accuracy decreases after distillation, as in the time-series experiments. However, for CNNs, the teacher model is stable, but the student model has poor accuracy and loss instability.

Table 5 lists the results of the ACC, the F1-Score, and the MCC of the teacher model and student model that were applied to the casting dataset.
Table 5. Casting dataset experiment results.

| Model   | ACC Top1 | F1-Score | MCC   |
|---------|----------|----------|-------|
| Teacher | CNN      | 94.79%   | 93.39%| 92.91%|
|         | AE       | 95.58%   | 94.12%| 93.72%|
| Student | CNN      | 90.02%   | 88.52%| 88.02%|
|         | AE       | 93.92%   | 92.47%| 92.05%|

In the case of the CNN, it can be seen that the accuracy of the student model is lowered. Since the CNN model shows low performance in the student model, on the basis of the alarm level, it is judged that secondary verification in the cloud is necessary.

Figure 9 shows the confusion metrics for the student model that was used in the fog. In the case of the AE, most defects and normals are classified well, but, in the case of the CNN, it is difficult to classify the defects and normals.

![Figure 9. Confusion metrics of the student model CNN and AE: (A) CNN; (B) AE.](image)

Table 6 shows the results of the second verification experiment according to the alarm level returned by the casting dataset using 100 samples of each randomly extracted test dataset.

Table 6. Alarm-level test result percentages for the casting dataset.

| Model             | Level 1 (%) | Level 2 (%) | Level 3 (%) |
|-------------------|-------------|-------------|-------------|
| Student (Fog) AE  | 88          | 2           | 10          |
| Teacher (Cloud) AE| 93          | 4           | 3           |
| Student (Fog) CNN | 68          | 5           | 27          |
| Teacher (Cloud) CNN| 91         | 3           | 6           |

Through the cloud secondary verification of the alarm level, the incorrect verifications were reduced by about 28%, compared to the fog-based classification.
5.3. Experiment and Results

In order to compare the experimental results that were obtained in the cloud and the fog, the experimental results of the two datasets were compared. Table 7 shows the experimental results of the model that was applied to the final CWRU dataset and to the casting dataset.

Table 7. Final experiment results.

| Model | ACC Top1 | F1-Score | MCC   |
|-------|----------|----------|-------|
| Teacher |          |          |       |
| LSTM  | 97.81%   | 96.32%   | 95.87%|
| GRU   | 95.52%   | 93.72%   | 93.22%|
| AE    | 95.58%   | 94.12%   | 93.72%|
| CNN   | 94.79%   | 93.39%   | 92.91%|
| Student |         |          |       |
| LSTM  | 96.77%   | 95.21%   | 94.76%|
| GRU   | 92.97%   | 91.37%   | 90.87%|
| AE    | 93.92%   | 92.47%   | 92.05%|
| CNN   | 90.02%   | 88.52%   | 88.02%|

The final results in Table 7 show good ACCs, F1-Scores, and MCC scores for the various cloud and fog models that were applied to either the time-series or image data. As expected, the teacher models always demonstrated an improvement over the student models. The LSTM model that was run on time-series data showed the highest performance, with both the teacher and student models showing very good performances.

6. Conclusions

In recent years, a great deal of research that is based on cloud–fog–edge technology has been conducted. With the development of hardware and communication technology, the roles and scopes of fog and edge applications are expanding. The existing cloud–fog–edge-based IIoT research pays more attention to communication and the implementation of lightweight algorithms. However, the knowledge distillation method, or the framework that has been proposed in previous studies, has several disadvantages that mean that it cannot be applied to heterogeneous equipment. The data that is collected from heterogeneous devices do not belong to a single type and are not of the same type. In this case, not only is it difficult to apply the algorithm to the field, but it is also necessary to recreate each algorithm according to the data type. Accordingly, this paper shows a better approach by proposing a framework that can be applied to heterogeneous equipment and that can compensate for the shortcomings of knowledge distillation. In this paper, the primary verification using the student model in the fog, and the secondary verification using the teacher model in cloud, were conducted. As a result of the experiment, the student model uses less memory and shows an accuracy of more than 90%. In other words, it is sufficient to apply to a computer with low computing power, but there is a problem with accurate judgment. To solve these shortcomings, a secondary verification process using the teacher model that is based on the alarm level was proposed. As a result of the experiment, the teacher model required a lot of computational power, but it showed a high accuracy of over 94%. In addition, in order to solve the bottleneck of the existing cloud network, we implemented a collaborative process that processes data, to some extent, in the fog through the alarm level.

The experiment was conducted with two datasets. It is necessary to experiment with the architecture that is proposed in this paper by using data that is collected from more heterogeneous equipment. Moreover, if the overall model performance is not good, there is the problem that the set alarm level cannot be used. However, it is expected that this approach can be readily extended to other IIoT classification and alarm problems.

In future work, we will collect data through more heterogeneous equipment, and we will study data-based alarm-level setting methods as well as a flexible framework that is applicable to more industrial sites by using various models. Moreover, research will be
conducted to reduce the response time faster in the proposed network, and research will be conducted to obtain numerical results according to the experiment.

**Author Contributions:** Conceptualization, S.O. and J.J.; methodology, S.O.; software, S.O.; validation, S.O.; D.K. and J.J.; formal analysis, S.O. and D.K.; investigation, S.O.; resources, J.J.; data curation, S.O.; writing—original draft preparation, S.O.; writing—review and editing, J.J.; visualization, S.O.; supervision, J.J. and C.L.; project administration, J.J.; funding acquisition, J.J. All authors have read and agreed to the published version of the manuscript.

**Funding:** This work was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICT Creative Consilience Program(IITP-2022-2020-0-01821) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation), and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2021R1F1A1060054).

**Acknowledgments:** This work was supported by the MSIT(Ministry of Science and ICT), Korea, under the ICT Creative Consilience Program(IITP-2022-2020-0-01821) supervised by the IITP(Institute for Information & communications Technology Planning & Evaluation), and the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (No. 2021R1F1A1060054).

**Conflicts of Interest:** The authors declare no conflict of interest.

**References**

1. Stergiou, C.; Psannis, K.E.; Kim, B.G.; Gupta, B. Secure integration of IoT and cloud computing. *Future Gener. Comput. Syst.* 2018, 78, 964–975. [CrossRef]
2. Gubbi, J.; Buyya, R.; Marusic, S.; Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. *Future Gener. Comput. Syst.* 2013, 29, 1645–1660. [CrossRef]
3. Shafieque, K.; Khawaja, B.A.; Sabir, F.; Qazi, S.; Mustaquit, M. Internet of things (IoT) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5G-IoT scenarios. *IEEE Access* 2020, 8, 23022–23040. [CrossRef]
4. Ahmed, E.; Yaqoob, I.; Gani, A.; Imran, M.; Guizami, M. Internet-of-things-based smart environments: State of the art, taxonomy, and open research challenges. *IEEE Wirel. Commun.* 2016, 23, 10–16. [CrossRef]
5. Sadiku, M.N.; Musa, S.M.; Momoh, O.D. Cloud computing: Opportunities and challenges. *IEEE Potentials* 2014, 33, 34–36. [CrossRef]
6. Dikaiakos, M.D.; Katsaros, D.; Mehra, P.; Pallis, G.; Vakali, A. Cloud computing: Distributed internet computing for IT and scientific research. *IEEE Internet Comput.* 2009, 13, 10–13. [CrossRef]
7. Wang, L.; Von Laszewski, G.; Younge, A.; He, X.; Kunze, M.; Tao, J.; Fu, C. Cloud computing: A perspective study. *New Gener. Comput.* 2010, 28, 137–146. [CrossRef]
8. Gomes, M.; da Rosa Righi, R.; da Costa, C.A. Internet of things scalability: Analyzing the bottlenecks and proposing alternatives. In Proceedings of the 2014 6th International Conference on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), St. Petersburg, Russia, 6–8 October 2014; pp. 269–276.
9. Jing, Q.; Vasilakos, A.V.; Wan, J.; Lu, J.; Qiu, D. Security of the Internet of Things: Perspectives and challenges. *Wirel. Netw.* 2014, 20, 2481–2501. [CrossRef]
10. Efremov, S.; Pilipenko, N.; Voskov, L. An integrated approach to common problems in the Internet of Things. *Procedia Eng.* 2015, 100, 1215–1223. [CrossRef]
11. Xingmei, X.; Jing, Z.; He, W. Research on the basic characteristics, the key technologies, the network architecture and security problems of the Internet of things. In Proceedings of the 2013 3rd International Conference on Computer Science and Network Technology, Dalian, China, 12–13 October 2013; pp. 825–828.
12. Muneeb, M.; Ko, K.M.; Park, Y.H. A Fog Computing Architecture with Multi-Layer for Computing-Intensive IoT Applications. *Appl. Sci.* 2021, 11, 11585. [CrossRef]
13. Xu, X.; Liu, Q.; Luo, Y.; Peng, K.; Zhang, X.; Meng, S.; Qi, L. A computation offloading method over big data for IoT-enabled cloud-edge computing. *Future Gener. Comput. Syst.* 2019, 95, 522–533. [CrossRef]
14. Liu, D.; Chen, X.; Zhou, Z.; Ling, Q. HierTrain: Fast hierarchical edge AI learning with hybrid parallelism in mobile-edge-cloud computing. *IEEE Open J. Commun. Soc.* 2020, I, 634–645. [CrossRef]
15. Pham, V.N.; Lee, G.W.; Nguyen, V.; Huh, E.N. Efficient Solution for Large-Scale IoT Applications with Proactive Edge-Cloud Publish/Subscribe Brokers Clustering. *Sensors* 2021, 21, 8232. [CrossRef] [PubMed]
16. Harmatos, I.; Malisoz, M. Architecture Integration of 5G Networks and Time-Sensitive Networking with Edge Computing for Smart Manufacturing. *Electronics* 2021, 10, 3085. [CrossRef]
17. Yun, D.W.; Lee, W.C. Intelligent Dynamic Real-Time Spectrum Resource Management for Industrial IoT in Edge Computing, *Sensors* 2021, 21, 7902. [CrossRef]
18. Wang, C.; Yang, G.; Papanastasiou, G.; Zhang, H.; Rodrigues, J.J.; de Albuquerque, V.H.C. Industrial cyber-physical systems-based cloud IoT edge for federated heterogeneous distillation. *IEEE Trans. Ind. Inform.* 2020, 17, 5511–5521. [CrossRef]
19. Moreno-Vozmediano, R.; Montero, R.S.; Llorente, I.M. Iaas cloud architecture: From virtualized datacenters to federated cloud infrastructures. Computer 2012, 45, 65–72. [CrossRef]
20. Jamsa, K. Cloud Computing: SaaS, PaaS, IaaS, Virtualization, Business Models, Mobile, Security and More; Jones & Bartlett Publishers: Burlington, MA, USA, 2012.
21. Pahl, C. Containerization and the paas cloud. IEEE Cloud Comput. 2015, 2, 24–31. [CrossRef]
22. Palos-Sanchez, P.R.; Arenas-Marquez, F.J.; Aguayo-Camacho, M. Cloud computing (SaaS) adoption as a strategic technology: Results of an empirical study. Mob. Inf. Syst. 2017, 2017, 2536040. [CrossRef]
23. Saifuzzaman, M.; Shetu, S.F.; Moon, N.N.; Nur, F.N.; Ali, M.H. IoT Based Street Lighting Using Dual Axis Solar Tracker and Effective Traffic Management System Using Deep Learning: Bangladesh Context. In Proceedings of the 2020 11th International Conference on Computing, Communication and Networking Technologies (ICCCNT), Kharagpur, India, 1–3 July 2020; pp. 1–5.
24. Hu, Y.C.; Patel, M.; Sabella, D.; Sprecher, N.; Young, V. Mobile edge computing—A key technology towards 5G. ETSI White Pap. 2015, 11, 1–16.
25. Premesan, G.; Di Francesco, M.; Taleb, T. Edge computing for the Internet of Things: A case study. IEEE Internet Things J. 2018, 5, 1275–1284. [CrossRef]
26. Sun, X.; Ansari, N. EdgeloT: Mobile edge computing for the Internet of Things. IEEE Commun. Mag. 2016, 54, 22–29. [CrossRef]
27. Yi, S.; Li, C.; Li, Q. A survey of fog computing: Concepts, applications and issues. In Proceedings of the 2015 Workshop on Mobile Big Data, Hangzhou China, 21 June 2015; pp. 37–42.
28. Bonomi, F.; Milito, R.; Zhu, J.; Addepalli, S. Fog computing and its role in the internet of things. In Proceedings of the first edition of the MCC workshop on Mobile cloud computing, Helsinki, Finland, 13–17 August 2012; pp. 13–16.
29. Dastjerdi, A.V.; Gupta, H.; Calheiros, R.N.; Ghosh, S.K.; Buyya, R. Fog computing: Principles, architectures, and applications. In Internet of Things; Morgan Kaufmann Publishers: Burlington, MA, USA, 2016; pp. 61–75.
30. Hinton, G.; Vinyals, O.; Dean, J. Distilling the knowledge in a neural network. arXiv 2015, arXiv:1503.02531.
31. Wang, J.; Yang, F.; Chen, T.; Shah, S.L. An overview of industrial alarm systems: Main causes for alarm overloading, research status, and open problems. IEEE Trans. Autom. Sci. Eng. 2015, 13, 1045–1061. [CrossRef]
32. Parsa, K.; Hassall, M.; Naderpour, M. Process Alarm Modeling Using Graph Theory: Alarm Design Review and Rationalization. IEEE Syst. J. 2020, 15, 2257–2268. [CrossRef]
33. Al-Kharaz, M.; Ananou, B.; Ouladsine, M.; Combal, M.; Pinaton, J. Evaluation of alarm system performance and management in semiconductor manufacturing. In Proceedings of the 2019 6th International Conference on Control, Decision and Information Technologies (CoDIT), Paris, France, 23–26 April 2019; pp. 1155–1160.
34. Shetu, S.F.; Saifuzzaman, M.; Moon, N.N.; Nur, F.N. A survey of Botnet in cyber security. In Proceedings of the 2019 2nd International Conference on Intelligent Communication and Computational Techniques (ICCT), Jaipur, India, 28–29 September 2019; pp. 174–177.