Anomaly Detection of Operating Equipment in Livestock Farms Using Deep Learning Techniques

Hyeon Park *, Daeheon Park and Sehan Kim

SDF Convergence Research Department, Electronics and Telecommunications Research Institute, Daejeon 34129, Korea; dhpark82@etri.re.kr (D.P.); shkim72@etri.re.kr (S.K.)
* Correspondence: hpark@etri.re.kr; Tel.: +82-42-860-5926

Abstract: In order to establish a smart farm, many kinds of equipment are built and operated inside and outside of a pig house. Thus, the environment for livestock (limited to pigs in this paper) in the barn is properly maintained for its growth conditions. However, due to poor environments such as closed pig houses, lack of stable power supply, inexperienced livestock management, and power outages, the failure of these environment equipment is high. Thus, there are difficulties in detecting its malfunctions during equipment operation. In this paper, based on deep learning, we provide a mechanism to quickly detect anomalies of multiple equipment (environmental sensors and controllers, etc.) in each pig house at the same time. In particular, environmental factors (temperature, humidity, CO$_2$, ventilation, radiator temperature, external temperature, etc.) to be used for learning were extracted through the analysis of data accumulated for the generation of predictive models of each equipment. In addition, the optimal recurrent neural network (RNN) environment was derived by analyzing the characteristics of the learning RNN. In this way, the accuracy of the prediction model can be improved. In this paper, the real-time input data (only in the case of temperature) was intentionally induced above the threshold, and 93% of the abnormalities were detected to determine whether the equipment was abnormal.

Keywords: anomaly detection; RNN; smart farming; oneM2M; environmental monitoring

1. Introduction

The scale of livestock farms has grown significantly and the number of livestock being reared is also increasing on a large scale. Farmers are looking for ways to increase pig productivity with a small number of staff. By paying more attention to livestock, we recognize that the health, well-being, and productivity of livestock can be improved [1,2]. This attention has led to the pursuit of so-called precision farming, and as part of its realization, interest in automated livestock smart farms is growing.

Recently, various services have been provided to enable precision agriculture to be implemented through IoT based platforms [3,4]. Thus, smart farming combined application of ICT solutions employed equipment such as environmental devices (i.e., IoT sensors, cameras, drones, robots, and so on) to deliver more productive farming.

Global livestock companies say that this smart agriculture solution keeps farms and livestock houses in excellent condition [5]. Therefore, in order to maintain an environment suitable for the growing conditions of livestock (limited to pigs in this paper), many kinds of equipment are built and operated inside and outside of barns barn.

To maintain such a suitable environment, many and various kinds of equipment are built and operated inside and outside of pig houses. It includes sensors that grasp the environment of pigs, such as temperature, humidity, CO$_2$, and ammonia sensors. And it also includes controllers that control those environments such as exhaust fans, flow fans, cooling pads, and radiators. Due to poor environments such as closed pig houses, lack of stable power supply, inexperienced livestock management, and power outages, the failure
of this equipment is high. However, there are difficulties in detecting its malfunctions during equipment operation.

In general, the operation of such equipment is mainly carried out by the initial installation and setup of each individual livestock farm. Thereafter, monitoring of the equipment is insufficient. Even with monitoring, systematic management and analysis of the collected data is not performed. Due to this, it is not possible to accurately and quickly detect malfunctions of the installed equipment, and a suitable environment is not maintained, which greatly affects pig productivity. Moreover, there are various types of livestock farms, and each of the pig houses has a variety of diverse equipment.

In this paper, we provide a mechanism to quickly detect abnormal situations of lots of equipment in each pig house at the same time. This mechanism includes a series of processes such as data collection in the pig houses, generation and distribution of models to predict malfunctions of various equipment.

First, in the case of data collection, data from lots of equipment installed in a pig house are collected. During the data collection process, livestock farms and equipment installed inside and outside the pig house have a client relationship with the data server. The data server collects data in real time through oneM2M.

Next, in order to generate an anomaly prediction model of various equipment, a learning environment for the prediction model is first established. That is, in order to control the temperature in the livestock house (for example, in the case of indoor temperature), in general, dozens of fans are adjusted in consideration of the indoor temperature, humidity, CO₂, cooling pad, radiator, and external temperature. The same is true for CO₂ and ammonia in the house. Determining whether these environmental factors operate normally is complex since they work organically with each other.

Complex systems are generally computationally expensive algorithms and are said to be the result of dynamics generated by the interaction of several subsystems [6,7]. As such, the livestock environment in this paper also belongs to a complex system, and prediction values are generated using huge data linked to each other (see Section 5.1.2). For this purpose, RNN, one of the deep learning techniques for time-sequential data analysis, was applied. Many sensors are installed inside and outside the pig house, and predictive models of each equipment must be performed at the same time. In order to increase the accuracy of the prediction model, the optimal RNN environment is derived by analyzing the characteristics of the learning RNN such as the RNN model type, the number of hidden layers, and the sequence number (see Section 3).

Finally, multi-equipment predicted models built through this learning are dynamically distributed to the livestock farm through tensorflow serving in the form of a client server. Tensorflow serving is a serving framework that can distribute predictive models. Clients such as livestock farm systems can easily handle model distribution since they can pass inputs to the model and receive results through the serving API. Thus, the mechanism simultaneously builds predictive models of multiple housing equipment anomalies using the docker container. It dynamically stores and distributes each model, and applies it to the livestock house using tensorflow serving. Due to a large computational overhead, each learning model is calculated by the central server, and the extracted prediction model is distributed to the farm. Based on the distributed model, each farmhouse generates predictive data whenever sensor and control data flows in real time to diagnose equipment malfunction.

1.1. Research Contributions

The contributions of this paper are as follows:

First, the environmental factors to be used for learning are extracted through the analysis of the accumulated data to create a predictive model of the equipment. Correlations between environmental factors by season for each equipment were analyzed. It was derived that the season has a close relationship with the environmental factors inside the barn. As such, the environment of the livestock is affected according to the season.
Thus, when learning data, season was included as an independent variable. The predicted model generated by this presented the predicted temperature, humidity, and CO\textsubscript{2} values according to the seasons.

Second, the optimal RNN environment was derived by analyzing the characteristics of the learning RNN. That is, only the most optimal elements were extracted by testing the number of cases such as RNN model type, number of hidden layers, and sequence number as a whole. By using this as a learning RNN environment, the accuracy of the predictive model can be increased.

1.2. Related Works

There were many previous works about management of livestock farming environment for livestock welfare and productivity. However, these studies include just simple control or remote monitoring of livestock barns for convenience.

Jianhua et al. [8] described a long-distance control system for livestock environments, and introduced wireless transmission techniques that can control equipment in livestock houses. They have also developed programs that can provide application services to smartphones. However, this paper is about the simple control of a pig house, and advanced technologies such as prediction and automatic control are not mentioned.

In Fancom’s white paper [9], it said that a freely adjustable interval ensures that hardly any feed is wasted. When the sow has eaten her complete ration, the system will stop dosing until the next feeding time. The number of feeding times can be set individually per sow. Initially, you set two feeding times a day. Later on, when the amount of feed increases, you can set up to eight feeding times a day. Although feeding can be flexibly adjusted within a given period, there is no mention of a technique that provides information by proactively predicting feeding.

In [10], to solve the problem of real-time remote monitoring of cage breeding environment an integrated program is implemented. The stability and adaptability testing of environmental monitoring equipment and systems was completed through the pilot deployment of IoT Ranch in a pigeon farm. However, it is also about the remote monitoring of cage breeding environment and advanced technologies such as prediction and automatic control are not mentioned.

In [11], they designed and implemented a system that detects, extracts, and analyzes objects of pig images in a livestock house to prevent diseases of livestock. They collected video information in the IoT environment and applied oneM2M to design and implement IoT client, IoT server, and functional architecture. However, it does not include an environment and a framework that can simultaneously monitor livestock diseases in several livestock, which is the main function of ICT-based for precision agriculture.

RNNs can be used for sequential data analysis [6,7], approximating the probability distribution that can calculate entropy measure using RNN. An industrial gas turbine (IGT) was considered to provide a case study for validation purposes. In this paper, we derive a prediction model that provides prediction values, approximating the RMSE of RNN (by Adam optimization method) by receiving the data of equipment in the livestock environment (input features) and performing learning.

1.3. Structure of the Paper

This paper is organized as follows. Section 2 introduces the proposed anomaly detection system. Section 3 explains how to generate the predictive models of equipment
in livestock house. Section 4 presents the application of the model to each farm. Section 5 describes the testbed environment including testbed building and learning and testing using data from livestock houses. Experimental results are also described in Section 5. Lastly, we make our conclusions in Section 6.

2. Anomaly Detection System

To address the proposed mechanism for the recognition of dynamic anomalies of multiple equipment in the pig house using deep learning techniques, it has a system structure as shown in Figure 1. Each pig house is built with IoT based environmental sensors and control equipment. In general, IoT refers to an intelligent network service that enables all things around us to be connected to the internet in order to communicate with each other and exchange information [13].

![Figure 1. System architecture for anomaly detection of operating equipment.](image)

In the IoT, communication between things occurs, and the underlying technology that supports this is machine-to-machine communication. However, in the case of such M2M, most of the developed IoT services and devices operate only in the same manufacturer and the same service area.

To solve such a problem, oneM2M, a standardized method between things in the internet of things, was proposed [14]. oneM2M is used in IoT devices as a standardized method for stable communication between devices [15]. It provides common functions such as remote configuration, operation instruction, connection, data collection, data storage, device management, and security [16]. oneM2M is a horizontal common platform, so it can reuse components implemented in software, regardless of service and industrial environments such as agriculture, logistics, medical care, automobiles, and home appliances.

In this paper, sensing and control information of these IoT-based devices is transmitted and received according to the oneM2M standard method.

On the other hand, the pig houses are equipped with a variety of different IoT equipment inside, depending on the purpose of pig houses. The equipment include temperature, humidity, CO2, ammonia sensor and exhaust fan, flow fan, cooling pad, radiator controller, etc., and generate data sensed and controlled data information both inside and outside the barn. Based on this information, in order to recognize the multi-equipment malfunction situation of livestock houses, first, it is necessary to learn the data collected to create a predictive model for each equipment.

Learning is done through the received and accumulated data, and these data are data collected from sensor equipment such as thermometers and hygrometers and control equipment such as exhaust fans in livestock houses.

These data are sensing values generated from actual devices and sensing values stored in oneM2M resources linked to sensor devices in the collection process. Therefore,
the oneM2M related database must be prepared and data of oneM2M must be stored regularly. Information of multiple devices installed in the livestock house must be collected periodically. This collection performs an organic linkage between the oneM2M device that provides information in the livestock house and the oneM2M service platform in the farm house regardless of the type or number of equipment in the livestock house. The information collected from the pig house is accumulated by data being provided to the central server to extract the predictive model of the multiple equipment of the pig house.

oneM2M standard entities used in this paper are oneM2M AE that includes application function logic to provide M2M service and oneM2M IN-CSE that provides common service functions that must be provided in common in oneM2M service platform. Data collection from livestock houses in the proposed system is considered under the oneM2M standard.

We also implemented a data adapter to accommodate any underlying data collection. The central server trains data regarding the relevant equipment of the livestock farm. Each trained model is stored in a model pool, and these models are distributed to each livestock farming system to be applied directly to the livestock farm. Based on this, when the data of the equipment are monitored in real time, these data are used to predict whether a malfunction of the equipment occurs. When an error of the threshold occurs within the critical period, information is provided to the user so that prompt action can be taken.

In this way, a series of tasks are performed dynamically: to collect data generated by each equipment of each livestock farm, learn each equipment data, store and distribute each model, and provide the result of determining the abnormal situation of each equipment. Based on this, when the data of the equipment are monitored in real time, these data are used to predict whether a malfunction of the equipment occurs. When an error of the threshold occurs within the critical period, information is provided to the user so that prompt action can be taken.

In this way, a series of tasks are performed dynamically: to collect data generated by each equipment of each livestock farm, learn each equipment data, store and distribute each model, and provide the result of determining the abnormal situation of each equipment. Based on this, when the data of the equipment are monitored in real time, these data are used to predict whether a malfunction of the equipment occurs. When an error of the threshold occurs within the critical period, information is provided to the user so that prompt action can be taken.

3. Predictive Model Generating

In this paper, we derive a prediction model that provides prediction values by receiving sensor data sequentially and learning. A livestock house can cause any change over time. The predictive model of each equipment is generated by performing learning in consideration of the previous occurrence data. Therefore, it is important to analyze the correlation of features to be input (see Section 3.1). Also, the temperature state (e.g., temperature case) applied to the previous RNN is not determined by one component of the room temperature. It is a state determined by the interaction of features (humidity, radiator, cooling pad, season, etc.) used as input. As such, how many accumulated states are read at a time provides an improved prediction function according to the sequence length or the RNN model type for controlling the memory cell during back propagation. In each section below, these contents are analyzed by actual experiments, and the results are used as an environment for learning.

3.1. Analysis of the Collected Data

In order to predict the anomaly of the equipment installed in the barn, it is necessary to create a predictive model for each equipment. For example, in the case of temperature (the indoor temperature of the livestock house), it is necessary to learn the previously accumulated data to make a predictive model. Rather than raising or lowering the temperature independently, it can be said that the temperature in the livestock house is dependent on other environmental factors such as the previous indoor temperature, the outside temperature, the ventilation amount of the exhaust fan, and the radiator temperature.

As such, it is necessary to analyze to extract dependent environmental factors to be used as independent variables when learning data for generating a temperature prediction model. Figure 2 shows which factors are related to temperature through Pearson’s coefficient of correlation graph.
The environmental factors included in the correlation were the previous indoor temperature, outside temperature, ventilation amount of exhaust fan, and radiator temperature that could affect the inside temperature of the livestock house. In winter and summer, factors other than internal temperature do not have much correlation with temperature.

On the other hand, in the case of spring and autumn, there is a positive correlation with the radiator temperature, and the radiator temperature has a negative correlation with the exhaust fan. As such, it seems that spring and summer have a relatively high correlation with the previous internal temperature, ventilation amount of exhaust fan, and radiator temperature.

In general, livestock farms do not operate exhaust fans much to keep the interior warm in winter, so the correlation with the ventilation amount of the exhaust fans seems to be small. In the case of summer, there is not much correlation because the radiators inside the barn are rarely operated.

Humidity is affected by the ventilation amount of the exhaust fan in winter, summer, and autumn as shown in Figure 3. On the other hand, the ventilation amount of the exhaust fan is affected by the previous internal humidity. In addition, spring and autumn are partially affected by external humidity and the amount of ventilation of the exhaust fan. In general, compared with temperature in Figure 2, it is understood that humidity is less affected by the ventilation amount of the exhaust fan.

As shown in Figure 4, winter, summer, spring, and autumn are affected by the previous CO$_2$, and unlike the temperature and humidity in Figures 2 and 3, it seems that the ventilation amount of the exhaust fan is insignificant. This means that even if the ICT system is well established in the livestock house, there is always a barn manager’s access. As a result, due to the high activity of pigs, CO$_2$ may increase, and the effect of ventilation may appear insignificant.
As such, the environment of the livestock house has an effect depending on the season, so when learning data, the season is included as an independent variable. The prediction model generated from this will present the predicted temperature, humidity, and CO$_2$ values according to the season.

Prior to prediction, based on the above analysis result, a formula for learning can be defined by using the equipment to be detected for malfunction and factors correlated with it. Learning creates an optimal predictive model through iterative calculations to reduce the difference between a hypothesis and an actual value based on previously accumulated data.

As shown in Equation (1), the hypothetically predicted temperature value (e.g., in the case of temperature) can be expressed as the addition of the error to the weight product of each of several independent elements (i.e., variables). That is, assuming the predicted temperature value $H(x)$, it corresponds to the independent variables $x$ such as the previous indoor temperature, the outside temperature, the ventilation amount of the exhaust fan, the radiator temperature, and the season. In particular, when there are many independent
variables that affect the predicted temperature, multi-variable actual values of \(x_1, x_2, x_3, x_4,\) and \(x_5\) are used as in Equation (2).

\[
H(x) = Wx + b \tag{1}
\]

\[
H(x_1, x_2, x_3, x_4, x_5) = w_1x_1 + w_2x_2 + w_3x_3 + w_4x_4 + w_5x_5 + b \tag{2}
\]

Equation (3) is a basic loss function for deriving a model by approximating the cross entropy. The predicted distribution value is calculated using the RMSE calculated as the difference between the hypothesis value \(H(x)\) predicted in Equation (2) above and the actual value \(y\). It will learn by receiving input repeatedly, which can be seen as the same as the probability distribution, which is a set of estimated probabilities.

\[
\text{cost}(W, b) = \frac{1}{m} \sum_{i=1}^{m} (H(x^{(i)}) - y^{(i)})^2 \tag{3}
\]

### 3.2. Review to Select the Optimal Neural Network Model

Based on the above correlation, deep learning RNN technique is used to diagnose anomaly of equipment in the livestock houses. Diagnosis of equipment malfunctions in livestock houses is difficult to determine as it is limited to a single point in time, and it should be analyzed in consideration of the previous series of time. Therefore, RNN deep learning technique for analyzing sequential data is appropriate. In addition, like the many-to-one method of RNN, an RNN deep learning technique as shown in Figure 5, which recurrently performs a previous series of time series data to generate one prediction value, is used.

![Figure 5. RNN schematic for sequential data analysis.](image)

Equation (4) shows the relationship between the new state and the old state in Figure 5 as a recurrence function. \(h_t\) is the new state, \(h_{t-1}\) is the old state, \(x_t\) is the input vector at some time step, and \(f_w\) is some function with parameters \(W\). Here, the state is the most basic vanilla RNN model of an RNN composed of a single hidden vector \(h\). By applying this recurrence formula at every time step, the sequence of vectors \(x\) can be processed. In this case, the same function and the same set of parameters are used in every time step.

\[
h_t = f_w(h_{t-1}, x_t) \tag{4}
\]

Equation (5) represents an equation considering the state update of Equation (4). \(W_{hh}\) is the weight from input to hidden, \(W_{hh}\) is from hidden to hidden, and \(W_{hp}\) is the weight from hidden to prediction \(y\). The current state of Equation (5), \(h_t\), is changed by the current input value and some history value. In general, in RNN, the \(f(x)\) function of the Equation (5) uses \(\tanh(x)\) a lot, and it can be written as Equation (6). After all, the hypothesis (prediction) \(y_t\) of Equation (3) can be expressed as Equation (7).

\[
h_t = f(W_{hh}h_{t-1}, W_{xh}x_t), \ y_t = f(W_{yh}h_t) \tag{5}
\]
\[ h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \]  
\[ y_t = W_{hy}h_t \]

In general, for RNN analysis, simple processing is performed on the input and hidden state like vanilla RNN according to Equation (5) to perform the neural net of the hidden layer. However, in RNN, sequential data needs to be processed, but if the sequence is long, \( \tanh \) eventually converge to 0 during back propagation and the gradient vanishes. As shown in the computational graph of Figure 6, when processing long sequences, continuous sequences must be processed as shown in Equations (8)–(10). However, continuous processing causes the accumulated \( \tanh \) to quickly approach 0 as in Equation (11), making learning difficult. Divergence can be achieved even if Relu is used instead of \( \tanh \). Also, \( y_t \) containing the hidden state changes radically.

**Figure 6.** Computational graph for processing of the long sequence data.

Thus, vanilla RNNs are difficult to train due to the gradient vanishing problem during back propagation, and thus the model accuracy is poor. Therefore, vanilla RNNs are vulnerable to learning long sequences. To solve this problem, LSTM with a gate mechanism, GRU, and the like have been proposed.

\[ h_t = \tanh(W[h_{t-1}, x_t]) \]  
\[ h_{t+1} = \tanh(W[h_t, x_{t+1}]) \]  
\[ h_{t+2} = \tanh(W[h_{t+1}, x_{t+2}]) \]  
\[ h_{t+n-1} = \tanh(W[tanh(\ldots tanh(\ldots h_{t-1}), x_{t+n-1})]) \]  

LSTM has a memory cell (cell state) that adds information flow to vanilla RNN as shown in Figure 7 and maintains necessary information during the sequence [17, 18]. By setting the cell state, \( y_t \) was changed gradually. The hidden state provides an output by appropriately processing the cell state. In order to control the memory cell, three gates are required, such as input, forget, and output gates.

**Figure 7.** LSTM for solving the banishing gradient problem.

The first forgetting gate is a gate for forgetting information from the past. As shown in Equation (12), it takes the hidden state \( h_{t-1} \) and \( x_t \), takes the sigmoid, and decides whether
to discard the previous state information [19]. In the input gate, in Equations (13) and (14), the sigmoid layer determines the value to be updated, and the \( \text{tanh} \) layer creates a vector \( \tilde{C}_t \) of new candidate values that can be added to the cell state. Through this process, Equation (15) \( C_t \) is created. The final step is the output gate, which determines what to output.

As shown in Equation (16), the sigmoid layer determines what information to output from the cell state, and puts the cell state into \( \text{tanh} \) so that the value has a value between \(-1\) and \(1\). Then, the \( \text{tanh} \) output is again multiplied by the sigmoid gate output so that only the part determined as in Equation (17) is output.

\[
\begin{align*}
    f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (12) \\
    i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (13) \\
    \tilde{C}_t &= \text{tanh}(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (14) \\
    C_t &= f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t \quad (15) \\
    o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (16) \\
    h_t &= o_t \cdot \text{tanh}(C_t) \quad (17)
\end{align*}
\]

GRU was first introduced in [20], and as shown in Figure 8, it is a modified structure from the LSTM to solve the gradient vanish problem while reducing the number of similar gates and parameters of the LSTM. There is no cell state in LSTM, only a hidden state. In Equations (18)–(20), gate \( r_t \), and \( z_t \) are made arbitrary \( h_t \), and as in Equation (21), eventually \( h_t \) is \( 1 - z_t \) and \( z_t \) so that \( h_{t-1} \) and \( h_t \) can be applied oppositely to each other. Although it has only two gates, an update gate and a reset gate, the operation speed is faster than the LSTM with three gates. In addition, as shown in the paper results in [21,22], it has similar or better performance than LSTM.

\[
\begin{align*}
    r_t &= \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (18) \\
    \tilde{h}_t &= \text{tanh}(W_c \cdot [r_t \cdot h_{t-1}, x_t]) \quad (19) \\
    z_t &= \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (20) \\
    h_t &= (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t \quad (21)
\end{align*}
\]

Figure 8. GRU: Simplified LSTM.

As such, it is important how the vanilla RNN, LSTM, and GRU result when applied to actual malfunction diagnosis. Gates are not a formula arranged by theory, but rather a belief that they will come out well by learning, so it is necessary to find out whether they are applied well in practice and to find the optimum for actual livestock operation. In this paper, as shown in the next section, the optimal RNN model type was selected based on the experimental results.
3.3. Extraction of Optimal RNN Elements

In this section, optimal learning factors are extracted to create a more accurate predictive model of each equipment for multi-equipment malfunction diagnosis. Here, the optimal learning factors include selecting hyper-parameters such as an appropriate hidden layer size or sequence length, an environmental factor to be measured, and an appropriate RNN model type. Spring (March to May), summer (June to August), autumn (September to November), winter (December to February) data and 10 temperature sensors, 4 humidity sensors, were extracted using 10 exhaust fans, 1 radiator, 1 CO2 sensor, and 1 outdoor temperature sensor outside the livestock house.

3.3.1. RNN Model Type

In the previous section, RNN model types were mentioned. Figure 9 compares how accurate these models are when generating predictive models for each equipment. The sequence length of the data was 7, the hidden layer was 5, the iteration was 2000 times, and the dropout was performed based on 0.2. As shown in Figure 9, each was performed for temperature, humidity, and CO2, and RMSE was calculated according to spring, summer, autumn, winter and all seasons. Although the difference in accuracy was insignificant in temperature, GRU was found to be higher than LSTM for humidity and CO2.

![Figure 9. Comparison of prediction models using RMSE of LSTM and GRU.](image)

3.3.2. Number of Hidden Layers

When performing deep learning such as RNN, how many hidden layers are used is important. This is due to the fact that complexity and performance may vary depending on the number of hidden layers.

When there are many layers, the performance is not high due to the high complexity, or the performance is not high since the number of layers is small. The performance may vary depending on the deep learning framework for generating the predictive model, so it can be derived by an experimental method. Figure 10 shows the prediction model accuracy according to the number of hidden layers of the GRU. When looking at temperature, humidity, and CO2, it can be seen that the accuracy is high when the total number of hidden layers is 5.

![Figure 10. Comparison of prediction models using RMSE according to number of hidden layers.](image)
3.3.3. Number of Training Sequence

A livestock house can cause any change over time, and in this paper, the prediction model of each equipment is created by learning in consideration of the previous generation data. This reflects the many-to-one analysis characteristics of RNN’s time series. In general, it is performed by taking the time sequence by max in the case of many to one, and it is not particularly determined.

However, as in this paper, when looking at the results of seasonal changes in livestock houses, it is expected that the accuracy of the prediction model will be affected depending on how much the previous time sequence is reflected and learning is performed. Therefore, in this paper, the test results of the prediction model according to the sequence length of the RNN were derived as shown in Figure 11.

![Figure 11. Comparison of prediction models using RMSE according to sequence length.](image)

3.3.4. Feature Cases

As mentioned in Section 3.1, it is necessary to collect the necessary element information when performing the learning of multiple equipment in the livestock house. However, not all factors mentioned in Section 3.1 were highly relevant to the predictive model to be obtained. In this paper, as shown in Table 1, when creating a predictive model of each equipment, test results are provided when all factors are applied and when only factors that are somewhat related are applied.

| Type of Sensor | Case #1 (All)                                                                 | Case #2                  |
|----------------|-------------------------------------------------------------------------------|--------------------------|
| **Temperature** | Season, Outside temperature, Fan control value, Radiator temperature, Inside temperature | Season, Fan control value, Radiator temperature, Inside temperature |
| **Humidity**   | Season, Outside humidity, Fan control value, Radiator temperature, Inside humidity | Season, Outside humidity, Inside humidity |
| **CO₂**        | Season, Fan control value, Inside CO₂                                         | Season, Fan control value, Inside CO₂ |

Judging from the experimental results as shown in Figure 12, it can be said that the accuracy is higher when a predictive model is made with generally related factors.
Figure 12. Comparison of prediction models using RMSE according to feature cases.

4. Application of the Model to Each Farm

4.1. Model Distribution

Figure 13 is a configuration diagram of the procedure for generating prediction models of multiple equipment in parallel in the central server and distributing them to livestock farms as local servers.

In order to recognize the anomaly of equipment in multiple livestock farms, the central server must perform a learning function for generating malfunction prediction models and a function of distributing the learned prediction models to the farms. For each piece of equipment, models of that equipment are continuously created over time. One of these models should be selected to test the data received from each device in real time.

As a result of the learning of each of the multiple devices in the livestock house, the learned models are actually generated from each device in real time, and are used to recognize the situation of malfunction.

In local servers, in order to detect the anomaly of the equipment, the models corresponding to each equipment must be accessed. In the server, an environment in the form of Docker Container [23] that can perform a task for accessing the predictive model for each equipment type is configured. In addition, tensorflow serving function is executed in each container to access the prediction model for prediction of equipment requested by livestock farms. At this time, the proposed system has a part of deep learning (parent class: PredictGraph) to be commonly executed when implementing the learning function of each device.

By inheriting this, a predictive model of equipment such as temperature, humidity, and CO₂ is implemented. When the class is initialized, deep learning network graph is initialized using hyper-parameter defined in code. It loads input data of each equipment (temperature, humidity, CO₂) to proceed with learning, and saves the learning result as a tensorflow saved model type to be loaded and served by tensorflow serving.

In livestock farms, the data of the equipment must be predicted by accessing the prediction model by receiving the indoor, outdoor environment sensing data and control data generated in real time in the livestock house. For this prediction, prediction is
performed by calling the Predict API to predict temperature, humidity, and CO2. At this time, since the current value of the tested data is predicted based on a previous series of data sequences, a series of input sequences and a result according to the prediction sequence are received in the form of output dimensions.

In order to predict anomaly of equipment that are actually received in real time, a prediction API in the form of RPC of tensorflow serving is used [24]. This API closely follows the PredictionService. Predict RPC API, and the request body of the predict API must be a JSON object.

4.2. Detailed Message Flow

Figure 14 shows a detailed block and the message flow of the dynamic anomaly detection. In the pig house, data generated from data client data, that is, temperature sensors, humidity sensors, carbon dioxide sensors, etc. (for example, temperature data tables) are identified and stored in the data server through the message broker of the local server. Data stored in the data server are transferred to the central server, delivered to each piece of equipment data for data training of the central server, accumulated in DB, and the generated models are stored in the model DB.

![Figure 14. Detailed block configuration and message flow for dynamic anomaly detection.](image)

Each model is performed with an instance, an object for each equipment’s recognition, which identifies the model of the equipment. Each equipment’s abnormal situation receives a request to run the model on the central server based on incoming data through their own model identifiers. These models are distributed to the relevant equipment model through a model distributor.

Through this, the model list of each equipment is stored as an analysis client in the local server of the livestock farm, and the model is used to make predictions through data input from the current data client. At this time, after determining the data correlation between the actual value and the predicted value, it is determined whether the equipment malfunctions according to the result.

The important factors for simultaneously detecting the abnormality of these multiple devices are as follows:
1. The central server continuously builds trained models for equipment using the accumulated data.
2. The trained models are stored in the model pool and dynamically distributed to livestock farms.
3. Livestock farms maintain information of distributed models (for example, model identification, etc.)
4. Livestock farmers acquire the predicted value through tensorflow serving as input of equipment data that is currently input in real time.

Here, in order to operate models of multiple devices as independent entities, a container is created as an instance for executing models for each device. The central server uses the tensorflow serving framework to store and distribute the trained model through this container operation. The livestock farm uses the tensorflow serving client to make predictions by calling the predictive model loaded on the tensorflow serving. The prediction request interface uses the Predict API provided by tensorflow serving.

5. Results
5.1. Testbed Environment
5.1.1. Building Test Bed

In this paper, as shown in Figure 15, a testbed was built in a livestock farm where pigs were raised to test whether equipment were abnormal.

In the pig house, sensors and controllers were installed in one piglet room as shown in Figure 15 in a closed shape, not exposed to the outside. Sensors include temperature, humidity, CO₂, ammonia, and controllers such as exhaust fans, wall fans, radiators, feeders, water heaters, and cooling pads.

Table 2 lists the sensor types inside and outside the test bed house used for testing. It indicates the purpose of use, product name, operating power, communication or power output, and the detailed location and purpose of use where these sensors are installed.

As shown in Figure 16, communication between the livestock farm and the office in the livestock farm is through LoRa. LoRa is a proprietary wireless communication standard developed by Semtech, which also provides excellent performance in terms of battery life. It is relatively inexpensive and resilient to transmission errors while maintaining wide coverage. Typical coverage ranges from 2–5 km to 20–25 km in urban areas [25].
Table 2. Sensor types inside and outside the test bed house used for testing.

| Location     | Category                     | Usage          | Product Name                  | Operating Power | Communication/Power Output | Detailed Location/Purpose | Amount |
|--------------|------------------------------|----------------|-------------------------------|-----------------|-----------------------------|---------------------------|--------|
| Inside Pig House | Sensor                     | Surface temperature | SA1-RTD-120 RTD              | RTD             | Piglet, Heating              | 2                         |        |
|              | Temperature                  | PR-20-2-100-1/4-2-E-T RTD | Piglet (9 in indoor, 1 on ceiling) | RTD             | Piglet                      | 10                        |        |
|              | Humidity (+Temperature)     | HTX75C-W-HT     | 12 V, 24 V Modbus RTU Piglet | Piglet          | 1                           |                           |        |
|              |                              | HTX75C-W-HT     | 12 V, 24 V Modbus RTU Aisle   | Piglet          | 1                           |                           |        |
|              |                              | HTX75C          | Mesh-Filter                  | Piglet, Aisle   | 8                           |                           |        |
|              | CO₂ (+Temperature/Humidity) | SH-VT260        | 12 V, 24 V Modbus RTU Piglet | Piglet, Need ventilation | 1                           |                           |        |
|              |                              | SH-VT260        | 12 V, 24 V Modbus RTU Aisle  | Piglet, Need ventilation | 1                           |                           |        |
| Outside Pig House | Meteorological observation equipment | Sensors | Davis Vintage Pro | Temperature/humidity | 1                           |                           |        |

Figure 16. Communication between livestock house and livestock farming offices.

Desktop computers and network switches are installed as ICT equipment in the offices of livestock farms. This desktop computer collects and stores equipment information in the pig house. It also operates as a client to access the malfunction prediction model of each device from the server. Office desktop computers access the central server through the internet and predict malfunctions of their farm equipment.

Figure 17 is a configuration diagram of a large number of various equipment inside and outside the livestock house that is subject to abnormality or malfunction detection, and a data collection layer diagram generated by the equipment. Most of the equipment in each pig house, such as piglets, sows, and breeding house, collects data generated through PLC.
Desktop computers and network switches are installed as ICT equipment in the offices of livestock farms. This desktop computer collects and stores equipment information in the pig house. It also operates as a client to access the malfunction prediction model of each device from the server. Office desktop computers access the central server through the internet and predict malfunctions of their farm equipment.

Figure 17 is a configuration diagram of a large number of various equipment inside and outside the livestock house that is subject to abnormality or malfunction detection, and a data collection layer diagram generated by the equipment. Most of the equipment in each pig house, such as piglets, sows, and breeding house, collects data generated through PLC.

Figure 17. Configuration diagram of a large number of various equipment and a data collection layer diagram.

The collected data is delivered through oneM2M-based protocol so that end users (Central Server (example: LIOS System)) can use it for appropriate application. Here, the oneM2M device is located in the collector installed in the corridor of the livestock house and delivers the data collected from the PLC to the oneM2M service platform. In the farm’s office, a DB that stores data collected through the oneM2M service platform is running. These accumulated data are used in the central server to learn data for deriving an equipment abnormal situation prediction model.

5.1.2. Learning and Testing

Figure 18 shows how to create a prediction model as one of the main components of the proposed mechanism. As shown in Figure 18a, RNN is performed to derive a learning model to predict the sensing or control value of each device.

The current indoor temperature is predicted by inputting a series of indoor temperature, previous temperature, outdoor temperature, ventilation amount, radiator temperature, and season, which are input data received from the livestock house. As one input of the predictive model, one line of training data in csv format consists of a total of 28 data because a feature set consisting of six data has a sequence consisting of seven. The input dimension is six, the output dimension is also four, and the value in the last column of the output result classification becomes the variable value for the actual predicted indoor temperature.

The data collected from 23 equipment in the barn in 4 seasons a year with a 5-min cycle are used as learning data as shown in Figure 18a. It shows the true (labeled data) Y of the room temperature (e.g., temperature) and the predicted (hypothesis) y (influencing factors are input features (humidity, radiator, season, etc.) as X). Here, Wh1h and Wh2h reflect the previous state, and the previous state is not a simple indoor temperature value, but a state value in which influencing factors are complexly calculated. These inputs are subjected to weighted hidden layers GRU to derive an output, and a trained model for prediction is extracted through iterative learning.
reflect the previous state, and the previous state is not a simple indoor temperature value, but a state value in which influencing factors are complexly calculated. These inputs are subjected to weighted hidden layers GRU to derive an output, and a trained model for prediction is extracted through iterative learning.

(a) RNN learning through accumulated equipment data

(b) Input data test using trained model of each equipment.

Figure 18. Training and testing through data of pig house equipment.

In this way, it is also applied to the generated values of other equipment installed in the actual livestock house, such as humidity, CO2, etc. If the sequence of X is 7 according to the RNN method, the next 8th value is predicted by using 7 values as inputs, so when the actual predicted value is obtained as shown in Figure 18a, it is acquired from the creation of the 7th value of x. In this form, it continuously accepts x data as inputs by 7 and performs training to optimize the estimated y value based on the labeled Y value, and a learning result, a learning model, is generated.

Figure 18b shows the process of predicting data of equipment (e.g., pig room temperature) in real-time using the model learned in Figure 18a. The input data is a series of data including currently occurring data, and the predicted data values are extracted through the trained mode.

Thus, the model learned is distributed to livestock farms to test incoming data in real time. As a result, the difference between the predicted value and the real-time value within a time period is monitored, and if it persists outside the threshold, an abnormal condition of the equipment is notified.
5.2. Experimental Results

This section describes the experimental results for room temperature, humidity, and CO2. We conducted the experiments on the Nvidia 1080 Ti server. The hyper-parameters for training were presented in Table 3. In Table 3, the hyper-parameter, learning rate, was tuned by random search [26]. Since the epoch of learning is generally around 2000 trials, this paper also tried this, and there was no difference in the accuracy of prediction from other epochs. Drop out was determined by the same procedure. As for the hidden layers and sequence length, 5 and 7 with good results were selected according to the test results in Section 3. In addition, in this paper, among the RNN model types, the GRU was selected according to the experimental results (see Section 3.3.1) considering the gradient vanishing problem and gates for controlling the memory cell during back propagation.

Table 3. Hyper-parameters for training.

| Hyper-Parameter   | Values |
|-------------------|--------|
| Learning Rate     | 0.01   |
| Epochs            | 2000   |
| Hidden Layers     | 5      |
| Sequence Length   | 7      |
| Drop out          | 0.2    |

Figure 19 is a picture showing the predicted value performance of sensor prediction models. It shows a graph comparing the actual measured value and the predicted value of the average of 9 indoor temperatures. As part of the 24-h measurement data, external temperature, total exhaust fan control, radiator temperature, and internal temperature were used as input to the RNN prediction model. The number of training data for making the prediction model, as shown Table 4, was 1,031,716, and during the test, the prediction mean error of 0.28 can be seen.

Figure 19. Graph comparing measured and predicted values (Indoor Temperature).

Table 4. The number of input data and features for training.

| Sensor                  | Input Features of RNN Prediction Model                                      | Training Data | Prediction Average Error |
|-------------------------|------------------------------------------------------------------------------|---------------|--------------------------|
| Temperature (9)         | Outside temperature, total control amount of exhaust fan, radiator temperature, indoor temperature | 1,031,716     | 0.28                     |
| Humidity (5)            | Total control amount of exhaust fan, Indoor humidity                        | 494,499       | 0.64                     |
| CO₂ (1)                 | total control amount of exhaust fan, Indoor CO₂                             | 130,326       | 4.73                     |

For humidity in Figure 20, the average value of five hygrometers was used, and the total amount of exhaust fan control and internal humidity were used as input to the RNN prediction model, the training data were 494,499 pieces and a prediction average error of
0.64 can be seen. The CO$_2$ in Figure 21 uses 1 CO$_2$ value, and the total exhaust fan control and internal CO$_2$ are used as input to the RNN prediction model. The training data are 130,326 pieces, and the prediction average error of 4.73 can be seen.

![Figure 20](image-url)  
**Figure 20.** Graph comparing measured and predicted values (Indoor Humidity).

![Figure 21](image-url)  
**Figure 21.** Graph comparing measured and predicted values (Indoor CO$_2$).

As shown in Figures 19–21, the number of malfunctions occurred zero times in temperature, zero times in humidity, and four times in CO$_2$. The sudden change in the actual measurement graph of CO$_2$ may be a malfunction of the sensor, and as it does not last for a long time, it is judged that a pig and an operator were working close to the CO$_2$ sensor location. This is a measured value as the concentration of CO$_2$ increases due to the movement of livestock and workers near the sensor installation using only one sensor value, and may differ from the predicted value by learning. To compensate for this, additional CO$_2$ sensors (e.g., at least four) are required to measure the CO$_2$ of the farm with an average value. Table 4 shows the data used for actual training and features of the equipment prediction model.

Although the diagnosis is continuously performed as shown in Figures 19–21, it is not easy to derive the performance of predictive models in a short period of time because pigs in the test bed are living. In this paper, considering this point, we intentionally caused real-time input data (in the case of temperature) to exceed the threshold, and monitored whether abnormal conditions of the equipment were recognized. Table 5 shows these results, and it can be seen that 93% of abnormal situations are detected.

| Number of Occurrences of Intentionally Real-Time Input Data (Temperature Case Only) | Number of Successful Diagnosis of Malfunctions | Number of False Diagnosis of Malfunctions |
|-----------------------------------------------------------------------------------|-----------------------------------------------|-------------------------------------------|
| 144                                                                               | 133 (93.3%)                                    | 11 (7.6%)                                  |
6. Conclusions

In this paper, we addressed the mechanism to predict the condition of the equipment currently being monitored based on the deployed model. A learning environment for each predictive model was built using RNN, one of the deep learning techniques, to predict the anomaly of each equipment. To operate many equipment in livestock houses, faster and more accurate models were built. For those models, the extraction process of environmental factors to be used for learning was addressed. In addition, the learning for the model differs in accuracy by the characteristics of the learning RNN such as the RNN model type, the number of hidden layers, and the sequence number were discussed. In this paper, an optimal RNN environment was derived by considering these points. These models are dynamically distributed to the pig house through tensorflow serving.

There is room for improvement by adding more information such as human access information and breeding management information to predict equipment malfunction. More precise use of the pig house environment will help to improve the performance of deep learning models with more training data. In addition, we plan to conduct research on livestock (for example, cattle) living in an open space. In this case, the house’s equipment is more affected by external temperature or humidity. Moreover, it is considered that the breakdown of equipment is also frequent due to frequent damage to electric wires and the like caused by wild animals (such as rats).

Author Contributions: Conceptualization, H.P. and S.K.; methodology and software, H.P.; data curation, D.P.; project administration, S.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No. 2018-0-00387, Development of ICT based Intelligent Smart Welfare Housing System for the Prevention and Control of Livestock Disease).

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

Abbreviations

The following abbreviations are used in this manuscript:

IoT Internet of Things
M2M Machine to Machine
ICT Information and Communication Technologies
AE Application Entity
IN-CSE Infrastructure Node—Common Service Entity
API Application Program Interface
LSTM Long-Short Term Memory
GRU Gated Recurrent Unit
RMSE Root-Mean Square Error
RPC Remote Procedure Call
JSON JavaScript Object Notation
DB Data Base
PLC Programmable Logic Controller
LoRa Long Range
LIOS Livestock Integration Operating System

References

1. Marcella, G.; Tomas, N.; Dries, B.; Erik, V.; Daniel, B. A blueprint for developing and applying precision livestock farming tools: A key output of the EU-PLF project. Anim. Front. 2017, 7, 12–17.
2. Erik, V.; Dries, B. Precision livestock farming for pigs. Anim. Front. 2017, 7, 1.
3. Ivan, A.; Craig, M.; Philippe, C.; Ahmed, J. Easy Global Market, Precision Livestock Farming Technologies. In Proceedings of the 2018 Global Internet of Things Summit (GIoTS), Bilbao, Spain, 4–7 June 2018.
4. Chunde, L.; Xianli, S.; Chuanwen, L. Edge Computing for Data Anomaly Detection of Multi-Sensors in Underground Mining. *Electronics* 2021, 10, 302.

5. Fancom. Total Automation Systems for Pigs. Available online: https://www.fancom.com/pigs (accessed on 1 June 2020).

6. Miguel, M.; Yu, Z.; Kenji, S.; Yudong, Z. Measuring System Entropy with a Deep Recurrent Neural Network Model. In Proceedings of the IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, 22–25 July 2019; pp. 1253–1256.

7. Miguel, M.; Yu, Z.; Kenji, S.; Yudong, Z. Deep Recurrent Entropy Adaptive Model for System Reliability Monitoring. *IEEE Trans. Ind. Informat.* 2021, 17, 839–848.

8. Jianhua, Z.; Fantao, K.; Zhifen, Z.; Shuqing, H.; Jing, Z.; Jianzhai, W. Development of Wireless Remote Control Electric Devices for Livestock Farming Environment. In Proceedings of the 2017 International Conference on Electronic Industry and Automation (EIA 2017), Suzhou, China, 23–25 June 2017; pp. 326–330.

9. Fancom. Automatic Feeding in the Farrowing House Is Worthwhile. Available online: https://www.fancom.com/white-papers/automatic-feeding-in-the-farrowing-house-is-worthwhile (accessed on 1 June 2020).

10. Xin, C.; Zhifen, Z.; Jianhua, Z.; Fantao, K.; Jianzhai, W.; Wei, S.; Xiangyang, Z.; Shuqing, H. Information Integration and Environmental Monitoring for Cage Pigeons. *IOP Conf. Ser. Earth Environ. Sci.* 2019, 371, 3.

11. Wonseok, J.; Hyeon, P.; SeHan, K.; Jeongwook, S. An IoT-Based Object Detection and Alerting System for Livestock Disease Prevention. In Proceedings of the 2019 International Conference on Future Information & Communication Engineering, Sapporo, Japan, 25–27 June 2019; pp. 337–340.

12. Ricardo, S.A.; Inés, S.; Oscar, G.; Javier, P.; Sara, R. An intelligent Edge-IoT platform for monitoring livestock and crops in a dairy farming scenario. *Ad. Hoc. Netw.* 2020, 98, 1.

13. Sehan, K.; Meonghun, L.; Changsun, S. IoT-Based Strawberry Disease Prediction System for Smart Farming. *Sensors* 2018, 18, 4051.

14. Onem2m TS-0001, Functional Architecture. August 2016. Available online: https://www.onem2m.org/images/files/deliverables/Release2/TS-0001-%20Functional_Architecture-V2_10_0.pdf (accessed on 5 January 2021).

15. Hyuncheol, P.; Hoichang, K.; Hotaek, J.; JaeSeung, S. Recent advancements in the Internet-of-Things related standards: A oneM2M perspective. *ICT Express* 2016, 2, 126–129.

16. Jorg, S.; Guang, L.; Philip, J.; Francois, E.; Jaeseung, S. Toward a standardized common M2M service layer platform: Introduction to oneM2M. *IEEE Wirel. Commun.* 2014, 21, 20–26.

17. Kwok, T.; Brij, B.; Pandian, V. A Genetic Algorithm Optimized RNN-LSTM Model for Remaining Useful Life Prediction of Turbofan Engine. *Electronics* 2021, 10, 285.

18. Bhargava, K.; Dursun, D. Predicting hospital readmission for lupus patients: An RNN-LSTM-based deep-learning methodology. *Comp. Bio. Med.* 2018, 101, 199–209.

19. Olach, C. Understanding LSTM Networks. 2015. Available online: http://colah.github.io/ (accessed on 16 June 2020).

20. Kyunghyun, C.; Bart, M.; Caglar, G.; Dzmitry, B.; Fethi, B.; Holger, S.; Yoshua, B. Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 21 October 2014; pp. 1724–1734.

21. Junyoung, C.; Caglar, G.; Kyunghyun, C.; Yoshua, B. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. In Proceedings of the 2014 NIPS Workshop on Deep Learning, Montreal, QC, Canada, 13 December 2014; pp. 1–9.

22. Seon-Min, L.; Young-Ghyu, S.; Jiyoung, L.; Donggu, L.; Eun-II, C.; Dae-Hyun, P.; Yong-Bum, K.; Isaac, S.; Jin-Young, K. Short-term Power Consumption Forecasting Based on IoT Power Meter with LSTM and GRU Deep Learning. *J. Inst. Internet Broad. Commun.* 2019, 19, 79–85.

23. Tensorflow Serving, Docker. Available online: https://www.tensorflow.org/tfx/serving/docker (accessed on 7 June 2020).

24. Tensorflow Serving, Client API. Available online: https://www.tensorflow.org/tfx/serving (accessed on 7 June 2020).

25. Lorenzo, G.; Vanni, M.; Giuseppe, B.; Luca, R.; Fabrizio, F. An IoT Architecture for Continuous Livestock Monitoring Using LoRa LPWAN. *Electronics* 2019, 8, 1435.

26. James, B.; Yoshua, B. Random search for hyper-parameter optimization. *J. Mach. Learn. Res.* 2012, 13, 281–305.