computation

MDPI

Article

A Study on Sample Size Sensitivity of Factory Manufacturing Dataset for CNN-Based Defective Product Classification

Dongbock Kim [1], Sat Byul Seo [1,*], Nam Hyun Yoo [2] and Gisu Shin [3]

[1] Department of Mathematics Education, Kyungnam University, 7 Kyungnamdaehak-ro, Masanhappo-gu, Changwon-si 51767, Gyeongsangnam-do, Korea
[2] School of Computer Science and Engineering, Kyungnam University, 7 Kyungnamdaehak-ro, Masanhappo-gu, Changwon-si 51767, Gyeongsangnam-do, Korea
[3] ANYTOY Co., Ltd., 59 Gwangyeocheonnam-ro, MasanHoewon-gu, MasanHoewon-gu, Changwon-si 51233, Gyeongsangnam-do, Korea
* Correspondence: sbseo@kyungnam.ac.kr; Tel.: +82-55-246-6445; Fax: +82-55-999-2149

Abstract: In many small- and medium-sized enterprises (SMEs), defective products are still manually verified in the manufacturing process. Recently, image classification applying deep learning technology has been successful in classifying images of defective and intact products, although there are few cases of utilizing it in practice. SMEs have limited resources; therefore, it is crucial to make careful decisions when applying new methods. We investigated sample size sensitivity to determine the stable performance of deep learning models when applied to the real world. A simple sequential model was constructed, and the dataset was reconstructed into several sizes. For each case, we observed its statistical indicators, such as accuracy, recall, precision, and F1 score, on the same test dataset. Additionally, the loss, accuracy, and AUROC values for the validation dataset were investigated during training. As a result of the conducted research, we were able to confirm that, with 1000 data points or more, the accuracy exceeded 97%. However, more than 5000 cases were required to achieve stability in the model, which had little possibility of overfitting.

Keywords: deep learning; defect classification; sample size sensitivity; convolutional neural network

check for updates

Citation: Kim, D.; Seo, S.B.; Yoo, N.H.; Shin, G. A Study on Sample Size Sensitivity of Factory Manufacturing Dataset for CNN-Based Defective Product Classification. *Computation* **2022**, *10*, 142. https://doi.org/10.3390/computation10080142

Academic Editors: Xiaoqiang Hua and Yudong Zhang

Received: 23 June 2022
Accepted: 16 August 2022
Published: 19 August 2022

1. Introduction

Rapid changes in technology and industry caused by increasing interconnectivity and smart automation have been defined as the Fourth Industrial Revolution (Industry 4.0). Automated manufacturing processes in factories are now in the spotlight under the Industry 4.0 paradigm and are actively implemented to accelerate productivity and efficiency in factories [1–3]. Smart manufacturing is widely discussed around the world and even small- and medium-sized enterprises (SMEs) are also considering potential changes by analyzing causes such as time-consuming processes, labor cost burdens, and management deterioration.

Quality inspection as part of the product manufacturing process is essential in every industry. One of the main product quality control processes is the classification of defective products in the manufacturing environment. However, this step in quality control is very time-consuming and labor-intensive because the inspection process is carried out manually; furthermore, its accuracy is not 100% due to human inefficiencies. This may therefore expose companies to large financial losses. These unreliable product classification systems bring challenges; for example, working in real-time with consistently high accuracy in difficult environments such as noisy real-world factories. Some methods have been utilized to address this task; among these, vision-based systems are well-known as very effective in classifying product quality in factories. Automated vision-based defect detection systems capable of distinguishing various kinds of materials, such as ceramics, textiles, and metals, have been studied [2,4].

AI-based quality inspection and defect detection systems are actively deployed by companies around the world. Recently, companies and factories are trying to change the inspection process to an automatic classification model based on deep learning. The latest AI technologies such as deep learning and computer vision have shown effectiveness in detecting various features in images with near-human accuracy. However, the majority of methods require a large amount of data. Applying AI techniques to real-world manufacturing is challenging due to the constraints associated with the need for large amounts of training data [5]. In real-world factories, especially small- and medium-sized enterprises (SMEs), it is more challenging to collect and label data. Smart manufacturing in SMEs faces several difficulties. First, a deep learning-based automated defect classification system requires data. AI models can provide an effective solution if production lines can generate a huge amount of data, but the solution may not be effective in others due to the limited amount of data [6]. SMEs generally have limitations in the manufacture of many products per day, which causes difficulties in collecting massive data for training a product dataset. In addition, developing an automated system in the product line is expensive and requires high domain expertise [6]. However, SMEs may not be able to cover the cost of smart factorization; thus, this should be determined before deciding whether the return on investment (ROI) of installing a camera or acquiring expertise for smart factorization is efficient.

Deep learning techniques have been applied in a variety of environments and have achieved remarkable results. In manufacturing environments, deep learning techniques have been used for computer vision and quality inspection tasks in order to classify and detect products [7–19]. Most of the research on deep learning applications in manufacturing has focused on a degree of performance using different pre-trained CNN models and different approaches to preprocessing. However, the first step for the initial AI factorization is usually to specify the data standards; this is being attempted in the actual manufacturing process in SMEs. Observing the performance changes in deep learning models according to the data size is essential for fields in which large amounts of data are difficult to obtain and process.

There have been studies evaluating the performance of deep learning models according to sample size changes in various fields [20–24]. Shahinfar et al. estimated the number of training images per animal species needed for a certain accuracy level [20]. Ladefoged et al. evaluated how training cohort size and input data affected the deep learning method in a clinical setup [21]. Barbedo investigated how the size of a dataset impacted the effectiveness of deep learning in plant pathology [22]. However, there have been no studies comparing performance depending on the change in sample size for manufacturing image data.

It is a quite obvious expectation that there should be a tendency toward better performance as the number of sample size increase for the stable model. However, most SMEs generally have limitations on manufacturing many products per day, which causes difficulties in collecting massive amounts of data for training a product dataset. This is the situation and the environment that many SMEs must confront. Thus, it is meaningful to find data size criteria for a guaranteed level of deep learning outcomes in SMEs.

In this study, we aim to specify the standards of dataset for an initial attempt of AI factorization in actual manufacturing process in SMEs, in which the difficulty of obtaining a large amount of data for deep learning training should be considered. We focused on the sample size sensitivity of a manufacturing image dataset and considered the number of samples required to stabilize a smart factory, comparing the model performance according to changes in the sample size. We used a simple sequential deep learning model and determined how many samples were needed to obtain a stable performance level for the deep learning model.

2. Materials and Methods

2.1. Dataset

To examine the sample size sensitivity on the CNN model in manufacturing image data, we used publicly available casting product image data from Kaggle, a platform on which data analysis competitions are held and preceded by the provision of data that requires analysis in terms of the company's research tasks and main services. The dataset from casting product image data for quality inspection is publicly available on Kaggle [25]. This dataset consists of top-view images of submersible pump impellers that are produced by casting. Many kinds of defect, such as burrs, pinholes, scratches, etc., can occur during casting; this dataset includes defective products and intact products.

2.1.1. Original Dataset from Kaggle

The original dataset on Kaggle contained 7348 images, and each image was 300 × 300 pixels in size, and grayscale. There are two categories: Defective front (def-front) and Ok front (ok-front). The number of training def-front images was 3758 and the number of training of-front images was 2875. In the test set, the number of def-front images was 453, and the number of ok-front images was 262. Figure 1 shows def-front and ok-front images. The dataset itself already included images that were augmented.

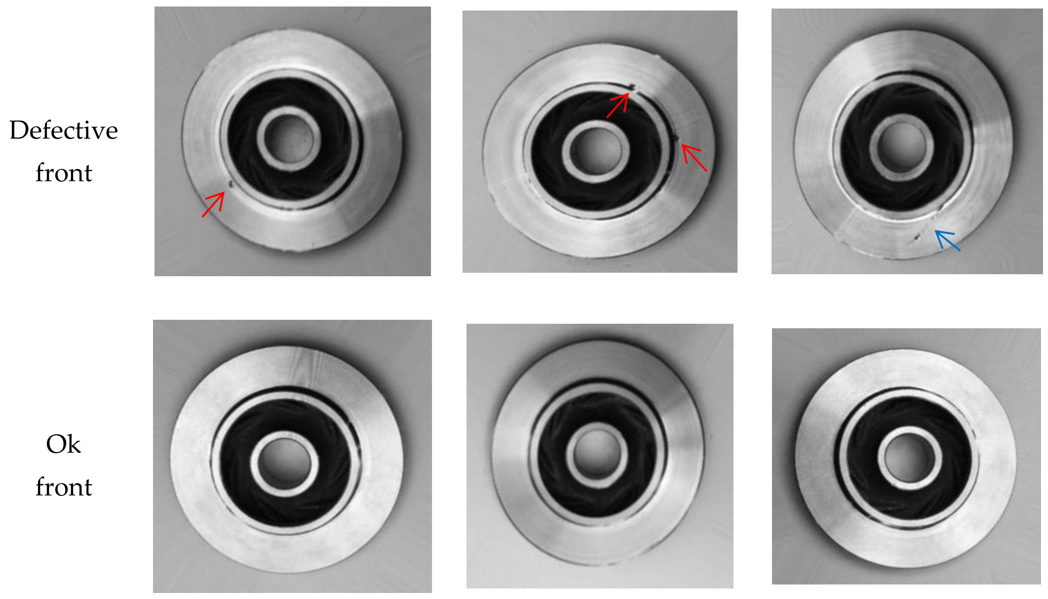


Figure 1. Def-front and ok-front samples from the dataset. Pinholes are marked with red arrows; scratches are marked with a blue arrow.

2.1.2. Reorganized Dataset

To compare the performance of CNN models according to the change in sample size, we built several different-sized datasets. We reorganized the datasets with 100, 300, 500, 700, 1000, 2000, and 5000 images, and trained models by each dataset. To monitor the learning process, we divided each dataset into training sets and validation sets in an 8:2 ratio with the same proportion of def-front and ok-front images. To evaluate the models, we applied the same test dataset to all models. The test dataset consisted of 715 samples. Our datasets, which were built from the original dataset, are summarized in Table 1.

Table 1. Information about the reorganized datasets.

Total Size	Train Size	Validation Size	Test Size
100	80	20	715
300	240	60	715
500	400	100	715
700	560	140	715
1000	800	200	715
2000	1600	400	715
5000	4000	1000	715

2.1.3. Data Preprocessing

The original dataset was collected for the purpose of data analysis; its background had already been normalized. Thus, in the preprocessing stage, a simple technique was applied. Each image of the dataset was in grayscale, and it was 300 pixels wide and 300 pixels high. We resized the images from 300 × 300 to 150 × 150 to improve the learning speed of the model. Before we applied data standardization, individual pixels ranged from 0 to 255 before preprocessing. To standardize, we calculated the mean standard deviation of the pixels and applied data standardization to our dataset.

Data standardization is an important element of data analysis [26]. Computers only perform mathematical operations with numbers; thus, serious problems can arise if the pixel values or the data values are severely different. For example, in most cases, 0 means black and 255 means white. There is a clear distinction between white and black for human recognition. However, if data are not standardized, computers could regard white as being much more important than black. In other words, when a computer calculates the values, white is treated as a more crucial factor than black because the value of white pixels is larger than that of black pixels. This can cause the model to run inaccurately. Figure 2 shows the images before and after preprocessing.

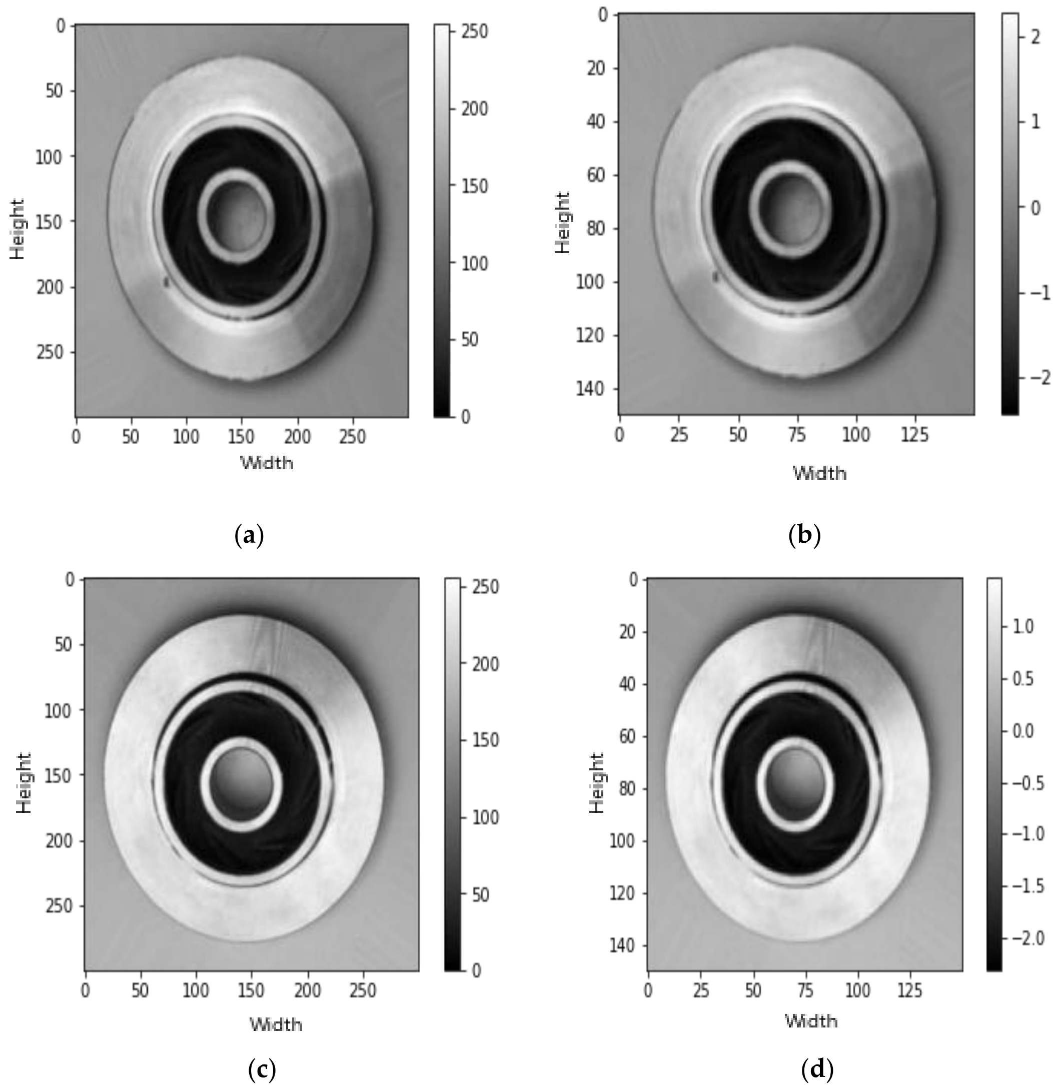


Figure 2. (**a**) def-front image before preprocessing; (**b**) def-front image after preprocessing; (**c**) ok-front image before preprocessing; (**d**) ok-front image after preprocessing.

2.2. Deep Learning

Machine learning techniques include artificial neural networks (ANNs) [27], which mimic structural aspects of the human nervous system. ANNs consist of an input layer, a hidden layer, and an output layer. If the ANN deals with two or more hidden layers, it is called deep learning or deep neural network.

2.2.1. Deep Neural Network

A typical deep neural network (DNN) is a kind of ANN that consists of several hidden layers. DNNs mainly have more than two hidden layers. Each layer consists of nodes with non-linear activation functions. When the data are entered through the input layer, the value of each node is multiplied by a weighting that determines the strength of the connection between the nodes, and a bias is added. In the second layer, the hidden layer, it passes through the activation function, which renders the DNN non-linear. By repeating this process, the numbers in the input layer are transferred to the output layer. This process is called feedforward.

In supervised learning, the answers for the input data are given. A DNN model produces its own output by performing a feedforward process. The output from the feedforward is a prediction of the model, and the error is the difference between the prediction and the given answer. Initially, an error is large because most of the initial weights and biases are set randomly. The model′s task is to reduce the errors. To do this, the model modifies its weights and biases. The final goal of the DNN is to make a prediction close to the given answer. In the learning process, the model has a loss function, and the learning proceeds in the direction of finding the minimum value of the loss function. To do this, a differential is required. If the loss is large enough, the loss function’s learning direction is decided by the partial differential coefficient of that loss. Using these differential coefficients, the weights and the biases are changed. As a result, the loss function can obtain smaller values. The process of finding the minimum value of the loss function by differentiation is called the gradient descent. Most DNN models use this method for optimization, and there are many kinds of gradient descent methods. Through this, the DNN parameters (weights and biases) are updated, which is called backpropagation [28]. By repeating feedforwards and backpropagations, the parameters are tuned and the loss is reduced. To achieve a meaningful performance, the model needs sufficient data. Fortunately, current supplies of data are quite smooth [29]. Thus, natural machine learning and deep learning have undergone considerable progress in recent years.

2.2.2. Convolutional Neural Network

Among deep learning technologies, convolutional neural networks (CNNs) are specialized in image recognition. CNN models have shown strong performance in the video and image recognition fields [30,31] and in the field of defect inspection [9,10]. CNNs represent a key change in deep learning technology. The basic principle and structure are similar to those of DNNs, but CNNs have several significant differences that distinguish them from general DNNs. First, when the input image enters, the CNN model identifies features of the image. To do this, the model uses filters that calculate the similarity of certain patterns. Filters represent small features of the image; clearly, the size of a filter is smaller than that of the original image. When initial random filters are given, each of them performs the convolution operation, which is a numeric calculation. By moving horizontally and vertically, the filter determines scores of the images. In this process, the weights in the filters change according to the small features of the image. This is called feature extraction. The images pass through these filters and become differently shaped images, which is called feature mapping. In other words, the filters represent what the CNN model learns during the learning. By training filters from input image data, CNN models can identify what is in the image [28].

2.3. CNN Architecture

CNN models mainly consist of three kinds of layers: convolution layers, pooling layers, and dense layers. In this study, we used a simple sequential model composed of twelve layers. The model consisted of four convolution layers, four max-pooling layers, two dense layers, a flattened layer, and a dropout layer, which was located directly before the dense layers to prevent overfitting.

All convolution layers used a 3 × 3 size filter with an activation function, a rectified linear unit (ReLU). Max-pooling layers, located directly after each convolution layer, used a 2 × 2-sized filter. We used binary cross-entropy as a loss function, and adaptive moment estimation (Adam) as an optimizer. Firstly, input images passed through the convolution layer and the max-pooling layer alternately four times and entered the flattened layer. Subsequently, they entered a dropout layer, used to prevent overfitting, and finally passed through two dense layers. They each had an activation function, ReLU and Sigmoid, respectively. We trained the models for 30 epochs and used a batch size of 16. The structure of the architecture is shown in Figure 3. Summaries of the layers and the parameters of the model are shown in Table 2.

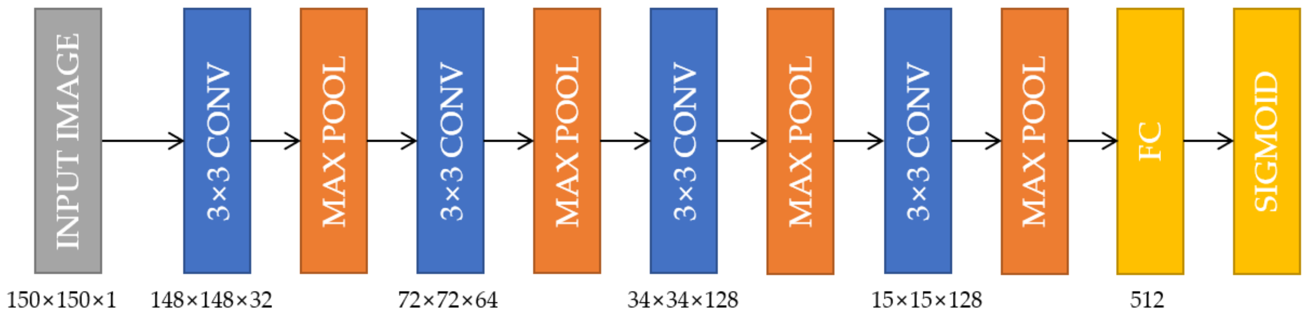


Figure 3. The structure of the architecture.

Table 2. Summary of the CNN model used with layers, output shapes, and the number of parameters.

Layer (Type)	Output Shape	Parameters
Conv2D(ReLU)	(None, 148,148,32)	320
MaxPooling2D	(None, 74,74,32)	0
Conv2D (ReLU)	(None, 72,72,64)	18,469
MaxPooling2D	(None, 36,36,64)	0
Conv2D (ReLU)	(None, 34,34,128)	73,856
MaxPooling2D	(None, 17,17,128)	0
Conv2D (ReLU)	(None, 15,15,128)	147,584
MaxPooling2D	(None, 7,7,128)	0
Flatten	(None, 6272)	0
Dropout	(None, 6272)	0
Dense (ReLU)	(None, 512)	3,211,776
Dense (Sigmoid)	(None, 1)	513
		Total: 3,452,545

2.4. Evaluation Metrics

Evaluation of the model is needed to confirm the model performance. In this section, we explain the evaluation metrics used in this research.

In binary classification (there are two actual classes, True and False): the model predicts the classes as True or False. Thus, there are four possible cases. The cases in our study were as follows: *TP*, *TN*, *FP*, and *FN*. True Positive (*TP*) meant that the model predicted the class as ok-front, and the real class was also ok-front. True Negative (*TN*) meant that the model predicted the class as def-front, and the answer was also def-front. False Positive (*FP*) meant that the model predicted the class as ok-front, but the answer was def-front. Finally, False Negative (*FN*) meant that the model predicted the class as def-front, but the answer was ok-front. A matrix made by these four cases is called a confusion matrix. We found some indicators that demonstrated the performance of the model using the cases mentioned above.

One of the indicators was Accuracy. This shows how correctly the model has made the predictions. The equation is as follows:

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{1}$$

Next, Precision was used to show the ratio of the True Positive class to that predicted as true by the model. Precision was calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

Recall (also: sensitivity) is the proportion between the True Positive class and the amount of the actual class that was true. The equation of recall is as follows:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

The *F*1 score is the harmonic mean of *Precision* and *Recall*. *Recall* and *Precision* are in a trade-off relationship; therefore, the *F*1 score shows how reliable the two indicators are. The *F*1 score can be expressed as follows:

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4}$$

Finally, a receiver operating characteristic (ROC) curve was used to determine the usefulness of the inspection model or to evaluate the accuracy of the model. ROC curves are an indicator mainly used to evaluate machine learning models. The area under the curve (AUC) can be calculated from the ROC curve. A high AUC value means that the accuracy of the inspection is high; the sensitivity and specificity must be known to derive the ROC curve. The sensitivity was consistent with the Recall, which is represented above. Specificity shows how thoroughly the model predicted the False class. The specificity equation is as follows:

$$Specificity = \frac{TN}{TN + FN} \tag{5}$$

2.5. Experimental Environment

We used an Intel Core i9-10900 X processor and GeForce RTX 2080 Ti. The dataset we used in this research was not too large; therefore, the hardware functioned without setbacks. Environment details are summarized in Table 3.

Table 3. Hardware and software environments.

Hardware Environment	Software Environment
CPU: Intel Core i9-10900 X, 3.7 GHz, Ten-core	Ubuntu TensorFlow 2.6.0 framework
Two threads, 32 GB	Python 3.7.6
GPU: Geforce RTX 2080 Ti	Keras 2.6.0

3. Results

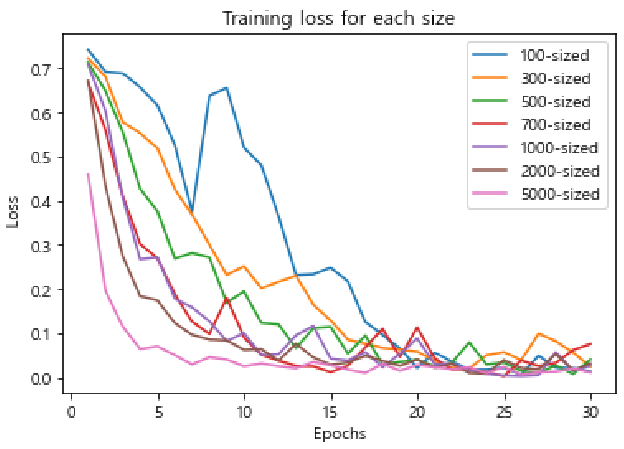
We built a simple CNN model to focus on how the model's performance changed with the sample size. The dataset we used in this research comprised image data of manufacturing facilities. We used the reorganized dataset and trained the models with a structurally identical architecture. We applied each model to the same test dataset and compared the evaluation indicators such as accuracy, precision, recall, and F1 score. A summary of the evaluation indicators of the n-sized dataset is shown in Table 4.

Table 4. A summary of the performance of each model.

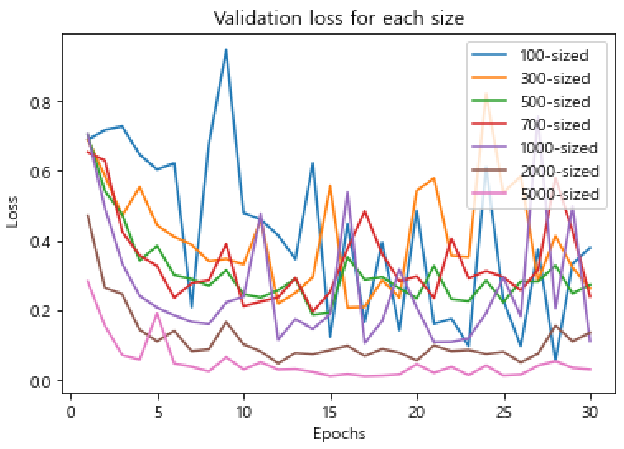
Dataset	Accuracy	Precision	Recall	F1 Score
100-sample	77.48%	66.56%	77.48%	71.6%
300-sample	88.95%	76.99%	99.62%	86.86%
500-sample	94.4%	87.5%	98.85%	92.83%
700-sample	95.94%	92.06%	97.33%	94.62%
1000-sample	98.32%	95.96%	99.62%	97.75%
2000-sample	97.76%	94.56%	99.61%	97.03%
5000-sample	98.46%	95.97%	100%	97.94%

As we expected, evaluation indicators showed better results as the sample size of the data increased, and the 5000-sample dataset was the highest among all datasets. However, the 1000-sample dataset demonstrated better performance in the test dataset than the 2000-sample dataset. However, it changed by random state, which decided how randomly the dataset was divided.

In the training process of each model, we compared the loss changes in the training set and the validation set. The graphs of loss change are shown in Figure 4. We found that the total number of samples with fewer than or equal to 1000 datasets were overfitted. The 2000-sample dataset was also slightly overfitted after the 28th epoch. Only the 5000-sample dataset exhibited stable loss change.



(**a**)



(**b**)

Figure 4. (**a**) Loss change in training set for each size; (**b**) loss change in validation set for each size.

We also compared the accuracy changes in the training set and the validation set. Two graphs are shown in Figure 5. The results for the accuracy changes in Figure 5 were similar to those in Figure 4.

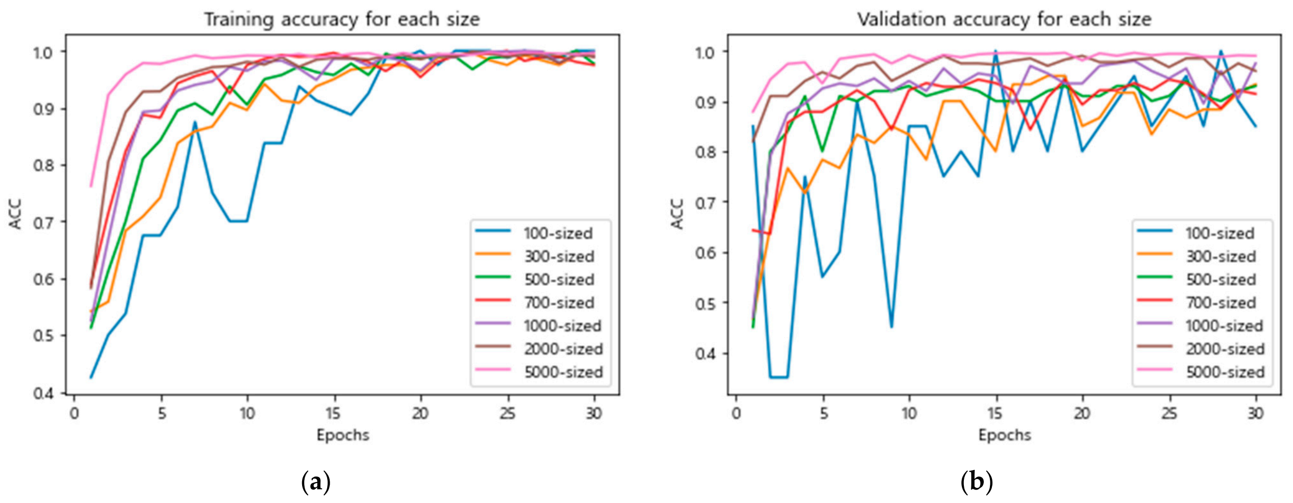


Figure 5. (**a**) Accuracy changes in training set for each size; (**b**) accuracy change in validation set for each size.

Figure 6 shows the ROC curves of each sample size. The AUC value, which means the area under the curve, ranges from 0.5 to 1; the closer to 1, the better the performance. The AUC value increases as the size of the dataset grows. As Figure 6 shows, the AUC value of the 5000-sample dataset reached 1.000, which is the optimum value of AUC.

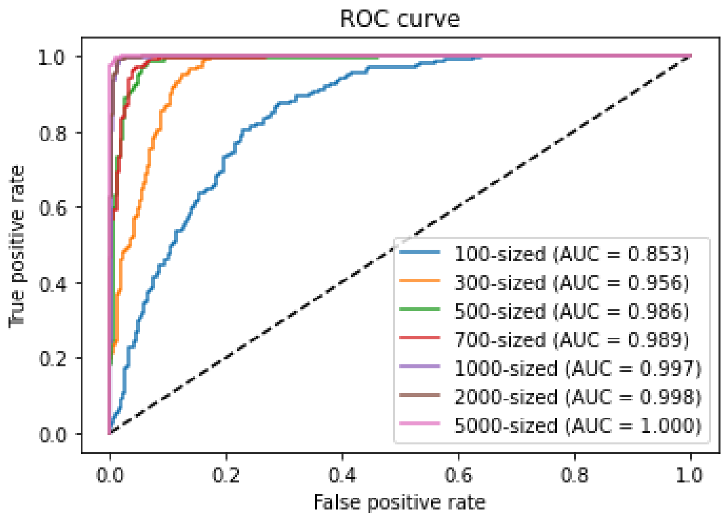


Figure 6. ROC curves and AUC values of the models for each sample size.

4. Discussion

Companies around the world actively conduct AI-based quality inspection and incorporate defect detection systems. Recently, companies and factories have been trying to change the inspection process towards automatic classification models based on deep learning. However, it is difficult to collect and label the data in real factories, especially small- and medium-sized enterprises (SMEs). SMEs generally have limitations in manufacturing many products daily and have difficulty collecting massive amounts of data for training product datasets. In addition, the development of an automated system within a production line is expensive and requires high domain expertise [6].

In this study, we aimed to find the appropriate sample size required to classify defective products in manufacturing data by evaluating the model performance according to sample size. We focused on the variability in model performance depending on the change in the sample size. We used a simple sequential deep learning model and identified how many samples were needed to stabilize the model performance. The loss and accuracy were monitored in each model, and we compared those results. As shown in Figures 4–6, it was confirmed that the model performance indicators were improved with an increasing sample size. The evaluation indicators showed better results as the sample size of the data increased, and the 5000-sample dataset was the best of all. However, we found that the total number of samples in datasets with fewer than or equal to 1000 samples was overfitted for the validation data, and that the 2000-sample dataset was also slightly overfitted after the 28th epoch. The 5000-sample dataset only exhibited stable loss changes. Therefore, we conclude that there was an overfitting issue in the models consisting of fewer than 5000 samples. Based on this model, overfitting issues would be solved when the sample size exceeds 5000. Thus, this implies that more than 5000 imaging samples are required to achieve a certain accuracy without introducing overfitting issues in the manufacturing field. In fact, Nguyen et al. evaluated the diagnostic performance of deep neural networks models using a dataset of over 7000 images, demonstrating approximately 98% accuracy [2].

Our research has some limitations. Firstly, we built a simple structure of the sequential model because we wanted to focus on only the data size sensitivity. Techniques for improving overfitting (regulations, early stopping, etc.) were not applied, and pre-trained models, which have been used in many other previous related studies, were not used. However, because we aimed to determine the ideal size of the dataset to be applied in real factories, we only focused on the variability in model performance in changing sample sizes. If transfer learning is applied, a better performance of the model is expected, even with smaller sample sizes. Secondly, the proportion of def-front images and ok-front images in the dataset was equal (1:1), in order to be balanced. However, in the real world, most of the data to be collected in factories are possibly imbalanced. In fact, the proportion of defective products might be significantly less than that of ok products. Obviously, the dataset can be constructed in a 1:1 ratio to match defective data, but as mentioned earlier, the better solution for SMEs is to recognize clear criteria in order to apply these deep learning technologies to the field. Therefore, comparing the performance of the models according to the ratio of defective data and acceptable data to find the best ratio seems worthy of further investigation. Thirdly, the dataset used in this study had little noise compared with the real factory data. Raw data from real factories are likely to be of lower quality compared with the casting data we used in this study. Hence, in order to construct an effective model using actual factory data, various preprocesses, including noise removal, may be required. Finally, our study did not seem to specify the application of a particular manufacturing process. Nevertheless, we aimed to specify the data standards necessary for initial AI factorization, which is being attempted in actual manufacturing processes. The data we used in this research were specific data; as mentioned above, we only used a model with the same structure. General problems of appearance defects at the manufacturing field are somewhat similar. Most types of defects in product images are burrs, pinholes, scratches, etc. Thus, generalization to a certain level may be possible even with specific data, but it seems that further research on this is needed.

5. Conclusions

Deep learning technologies are developing rapidly, and this tendency will continue for decades to come. Despite these advances, many SMEs have trouble applying this technology in real fields. In this study, we focused on the size sensitivity of the manufacturing image data. Thus, we reorganized the whole dataset. The CNN model with the same structure was applied to each dataset. To observe the result according to the datasets, we monitored the loss function and the accuracy changes in the learning process; several statistical metrices such as accuracy, precision, recall, and F1 score were monitored using the test dataset. We aimed to find the sample size criterion required for the CNN-based classification of defective products in manufacturing data for application in practical factories, especially SMEs.

Author Contributions: S.B.S., N.H.Y. and G.S. conceived the projects; D.K. and S.B.S. designed the experiments and organized the manuscript; D.K. developed neural network architectures and performed computational experiments; D.K. and S.B.S. discussed the experimental results and wrote the manuscript; S.B.S., D.K., N.H.Y. and G.S. reviewed and edited the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Establishment of Demonstration Infrastructure for Regulation-Free Special Zones funded by the Ministry of SMEs and Startups (MSS, Korea).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: [https://archive.ics.uci.edu/ml/datasets.php], accessed on 22 June 2022.

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

References

1. Büchi, G.; Cugno, M.; Castagnoli, R. Smart factory performance and Industry 4.0. *Technol. Forecast. Soc. Chang.* **2020**, *150*, 119790. [CrossRef]
2. Nguyen, H.; Yu, G.-H.; Shin, N.-R.; Kwon, G.-J.; Kwak, W.-Y.; Kim, J.-Y. Defective Product Classification System for Smart Factory Based on Deep Learning. *Electronics* **2021**, *10*, 826. [CrossRef]
3. Osterrieder, P.; Budde, L.; Friedli, T. The smart factory as a key construct of industry 4.0: A systematic literature review. *Int. J. Prod. Econ.* **2020**, *221*, 107476. [CrossRef]
4. Czimmermann, T.; Ciuti, G.; Milazzo, M.; Chiurazzi, M.; Roccella, S.; Oddo, C.M.; Dario, P. Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A Survey. *Sensors* **2020**, *20*, 1459. [CrossRef]
5. Deshpande, A.M.; Telikicherla, A.K.; Jakkali, V.; Wickelhaus, D.A.; Kumar, M.; Anand, S. Computer Vision Toolkit for Non-invasive Monitoring of Factory Floor Artifacts. *Procedia Manuf.* **2020**, *48*, 1020–1028. [CrossRef]
6. Kang, Z.; Catal, C.; Tekinerdogan, B. Machine learning applications in production lines: A systematic literature review. *Comput. Ind. Eng.* **2020**, *149*, 106773. [CrossRef]
7. Adibhatla, V.A.; Chih, H.-C.; Hsu, C.-C.; Cheng, J.; Abbod, M.F.; Shieh, J.-S. Defect Detection in Printed Circuit Boards Using You-Only-Look-Once Convolutional Neural Networks. *Electronics* **2020**, *9*, 1547. [CrossRef]
8. Le, N.T.; Wang, J.-W.; Shih, M.-H.; Wang, C.-C. Novel Framework for Optical Film Defect Detection and Classification. *IEEE Access* **2020**, *8*, 60964–60978. [CrossRef]
9. Li, C.; Zhang, X.; Huang, Y.; Tang, C.; Fatikow, S. A novel algorithm for defect extraction and classification of mobile phone screen based on machine vision. *Comput. Ind. Eng.* **2020**, *146*, 106530. [CrossRef]
10. Liong, S.-T.; Zheng, D.; Huang, Y.-C.; Gan, Y.S. Leather defect classification and segmentation using deep learning architecture. *Int. J. Comput. Integr. Manuf.* **2020**, *33*, 1105–1117. [CrossRef]
11. Lu, M.; Mou, Y. Bearing Defect Classification Algorithm Based on Autoencoder Neural Network. *Adv. Civ. Eng.* **2020**, *2020*, 6680315. [CrossRef]
12. Nguyen, H.T.; Shin, N.-R.; Yu, G.-H.; Kwon, G.-J.; Kwak, W.-Y.; Kim, J.-Y. Deep learning-based defective product classification system for smart factory. In Proceedings of the 9th International Conference on Smart Media and Applications, Jeju-si, Korea, 17–19 September 2020. [CrossRef]
13. Nguyen, T.P.; Choi, S.; Park, S.-J.; Yoon, J. Inspecting Method for Defective Casting Products with Convolutional Neural Network (CNN). *Int. J. Precis. Eng. Manuf. Technol.* **2021**, *8*, 583–594. [CrossRef]
14. Park, J.; Riaz, H.; Kim, H.; Kim, J. Advanced cover glass defect detection and classification based on multi-DNN model. *Manuf. Lett.* **2020**, *23*, 53–61. [CrossRef]
15. Tello, G.; Al-Jarrah, O.Y.; Yoo, P.D.; Al-Hammadi, Y.; Muhaidat, S.; Lee, U. Deep-Structured Machine Learning Model for the Recognition of Mixed-Defect Patterns in Semiconductor Fabrication Processes. *IEEE Trans. Semicond. Manuf.* **2018**, *31*, 315–322. [CrossRef]
16. Wang, J.; Fu, P.; Gao, R.X. Machine vision intelligence for product defect inspection based on deep learning and Hough transform. *J. Manuf. Syst.* **2019**, *51*, 52–60. [CrossRef]
17. Yang, Y.; Lou, Y.; Gao, M.; Ma, G. An automatic aperture detection system for LED cup based on machine vision. *Multimed. Tools Appl.* **2018**, *77*, 23227–23244. [CrossRef]
18. Yun, J.P.; Shin, W.C.; Koo, G.; Kim, M.S.; Lee, C.; Lee, S.J. Automated defect inspection system for metal surfaces based on deep learning and data augmentation. *J. Manuf. Syst.* **2020**, *55*, 317–324. [CrossRef]
19. Zhang, E.; Li, B.; Li, P.; Chen, Y. A Deep Learning Based Printing Defect Classification Method with Imbalanced Samples. *Symmetry* **2019**, *11*, 1440. [CrossRef]
20. Shahinfar, S.; Meek, P.; Falzon, G. "How many images do I need?" Understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring. *Ecol. Inform.* **2020**, *57*, 101085. [CrossRef]

21. Ladefoged, C.N.; Hansen, A.E.; Henriksen, O.M.; Bruun, F.J.; Eikenes, L.; Øen, S.K.; Karlberg, A.; Højgaard, L.; Law, I.; Andersen, F.L. AI-driven attenuation correction for brain PET/MRI: Clinical evaluation of a dementia cohort and importance of the training group size. *NeuroImage* **2020**, *222*, 117221. [CrossRef]
22. Barbedo, J.G.A. Impact of dataset size and variety on the effectiveness of deep learning and transfer learning for plant disease classification. *Comput. Electron. Agric.* **2018**, *153*, 46–53. [CrossRef]
23. Cho, J.; Lee, K.; Shin, E.; Choy, G.; Do, S. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv* **2015**, arXiv:1511.06348.
24. Ng, W.; Minasny, B.; Mendes, W.D.S.; Demattê, J.A.M. The influence of training sample size on the accuracy of deep learning models for the prediction of soil properties with near-infrared spectroscopy data. *Soil* **2020**, *6*, 565–578. [CrossRef]
25. Available online: https://www.kaggle.com/datasets/ravirajsinh45/real-life-industrial-dataset-of-casting-product (accessed on 10 January 2022).
26. Shanker, M.; Hu, M.; Hung, M. Effect of data standardization on neural network training. *Omega* **1996**, *24*, 385–397. [CrossRef]
27. Wang, S.-C. Artificial neural network. In *Interdisciplinary Computing in Java Programming*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 81–100.
28. Chollet, F. *Deep Learning with Python*; Simon and Schuster: New York, NY, USA, 2021.
29. Duan, Y.; Edwards, J.S.; Dwivedi, Y.K. Artificial intelligence for decision making in the era of Big Data–evolution, challenges and research agenda. *Int. J. Inf. Manag.* **2019**, *48*, 63–71. [CrossRef]
30. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet classification with deep convolutional neural networks. *Adv. Neural Inf. Processing Syst.* **2012**, *25*. [CrossRef]
31. Lecun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]