mathematics

MDPI

Article

Defect Detection Model Using CNN and Image Augmentation for Seat Foaming Process

Nak-Hun Choi [1], Jung Woo Sohn [2] and Jong-Seok Oh [1,3,*]

[1] Department of Future Convergence Engineering, Kongju National University, Cheonan 31080, Chungnam, Republic of Korea; nnk_0950@naver.com
[2] Department of Mechanical Design Engineering, Kumoh National Institute of Technology, Gumi 39177, Gyeongbuk, Republic of Korea; jwsohn@kumoh.ac.kr
[3] Department of Future Automotive Engineering, Kongju National University, Cheonan 31080, Chungnam, Republic of Korea
* Correspondence: jongseok@kongju.ac.kr

Abstract: In the manufacturing industry, which is facing the 4th Industrial Revolution, various process data are being collected from various sensors, and efforts are being made to construct more efficient processes using these data. Many studies have demonstrated high accuracy in predicting defect rates through image data collected during the process using two-dimensional (2D) convolutional neural network (CNN) algorithms, which are effective in image analysis. However, in an environment where numerous process data are recorded as numerical values, the application of 2D CNN algorithms is limited. Thus, to perform defect prediction through the application of a 2D CNN algorithm in a process wherein image data cannot be collected, this study attempted to develop a defect prediction technique that can visualize the data collected in numerical form. The polyurethane foam manufacturing process was selected as a case study to verify the proposed method, which confirmed that the defect rate could be predicted with an average accuracy of 97.32%. Consequently, highly accurate defect rate prediction and verification of the basis of judgment can be facilitated in environments wherein image data cannot be collected, rendering the proposed technique applicable to processes other than those in this case study.

Keywords: defect detection; defect prediction; manufacturing process; seat foaming process; deep learning; convolutional neural network; image augmentation; artificial neural network

MSC: 37M20

Citation: Choi, N.-H.; Sohn, J.W.; Oh, J.-S. Defect Detection Model Using CNN and Image Augmentation for Seat Foaming Process. *Mathematics* **2023**, *11*, 4894. https://doi.org/10.3390/math11244894

Academic Editor: Konstantin Kozlov

Received: 18 November 2023
Revised: 4 December 2023
Accepted: 5 December 2023
Published: 7 December 2023

1. Introduction

Recently, the "4th Industrial Revolution" has emerged as a major keyword for economic growth and has had a great effect in various fields, including manufacturing. In particular, the concept of Industry 4.0, which enables factories to become independent and self-adaptive depending on input from data that are gathered, is known in manufacturing as smart manufacturing. Smart factories are production systems wherein factory devices and parts are connected and interact with each other by combining existing production manufacturing technologies with technologies such as the Internet of things, big data, and cloud computing. A key feature of smart manufacturing is to assess and extract relevant information from collected data using deep learning [1–3].

Because deep learning can analyze raw data and automatically identify feature representations of data across several levels of abstraction, it has attracted interest as a tool in smart manufacturing. The application of deep learning is not limited to process fault monitoring [4–6] or state estimation [7,8]; several studies have explored its potential for various other manufacturing applications [9,10]. In deep learning, artificial neural networks (ANNs) and convolutional neural networks (CNNs) are widely acknowledged as the leading technologies for pattern recognition from tabular and image data, respectively.

Each layer of an ANN is made up of a collection of several perceptrons or neurons. Because an ANN only processes inputs in a forward manner, it is often referred to as a feedforward neural network. It can easily be used to process image, textual, and tabular data. Such neural networks are among the simplest variants. They pass information in one direction through various input nodes until sending it to an output node. The network may or may not have hidden node layers, rendering their functions more interpretable. Several studies have shown that ANNs can implicitly detect complex nonlinear relationships between dependent and independent variables. However, proper feature selection is crucial when applying an ANN. The features input into the model must be well designed according to the problem at hand. A CNN comprises convolution, pooling, and fully connected layers. A CNN is best used when millions of features need to be retrieved, since the convolutional layer generates feature maps that capture an area of an image that is then divided into rectangles and transmitted for nonlinear processing. The CNN automatically aggregates these characteristics rather than measuring each one separately. The fully connected layers use the extracted features to identify the input picture after the pooling layer reduces the number of the collected features [11,12]. CNNs based on auto-feature extraction have been used in various systems for fault detection, material degradation, and other applications. Glaeser et al. developed a fault-detection algorithm for industrial cold forging. Based on a CNN, the algorithm can detect faults with 99.02% accuracy, and classify each fault with 92.66% accuracy [13]. Nakazawa and Kulkarni proposed a CNN with a SoftMax activation function to classify 22 WM defect patterns [14]. Saqlain et al. proposed a deep learning-based CNN for automatic wafer defect identification (CNN-WDI) in semiconductor manufacturing processes [15]. However, CNNs are better suited for processing image data rather than tabular data. Accordingly, several studies have utilized the conversion of tabular data into image data to leverage the advantages of CNNs, such as automatic feature extraction. Numerous time–frequency analysis techniques, including short-time Fourier transform (STFT), continuous wavelet transform (CWT), and wavelet packet transform (WPT), were combined with CNNs to convert tabular time-series data [16]. These techniques use deep learning techniques to extract discriminative features from time–frequency representations rather than the time domain and convert continuous time-series data to two-dimensional (2D) representations using time–frequency analysis [17–21]. The second method involves the conversion of numbers into images for noncontinuous time-series data. Sezer et al. [22] generated 15 × 15 pixel images using 15 technical indicators related to stock prices. A CNN was adopted as the classification and prediction model to classify financial data as images and predict buy, sell, or hold signals for stocks. They evaluated the performance of their proposed model on Dow 30 stocks. In addition, Lee et al. [23] converted tabular data, such as vehicle spare parts, into 3D voxel images and applied them to a 3D CNN to perform demand forecasting for spare parts. By comparing them with other methods, they concluded that the proposed method exhibited good prediction performance. However, there has been no research related to the application of CNNs using the dataset conversion of numbers into images for manufacturing processes. In addition, many manufacturing process data are recorded as noncontinuous time-series and tabular data types.

Consequently, the main contribution of this study is to detect defects in manufactured products by applying data obtained from the seat foam manufacturing process to the CNN algorithm. Since the data obtained from the manufacturing process are numerical data in tabular form, they were normalized, converted to gray images, and applied to the CNN algorithm. To solve the imbalanced data problem, data augmentation and hyperparameter optimization were also performed. In order to confirm the excellence of the proposed method, defect detection was performed by applying the features extracted from tabular numerical data to the ANN algorithm and then comparing the results with the results of the proposed method. Consequently, it was possible to develop a defect detection model with an accuracy of 98.33%, and the results confirmed the effectiveness of the proposed technique.

2. Data Collection and ANN

2.1. Data Collection and Processing Method

The validity of this study was verified using a polyurethane foaming process for automobile seats. The manufacturing equipment used for the foaming process is shown in Figure 1. The study was conducted using data recorded during the actual foaming process.

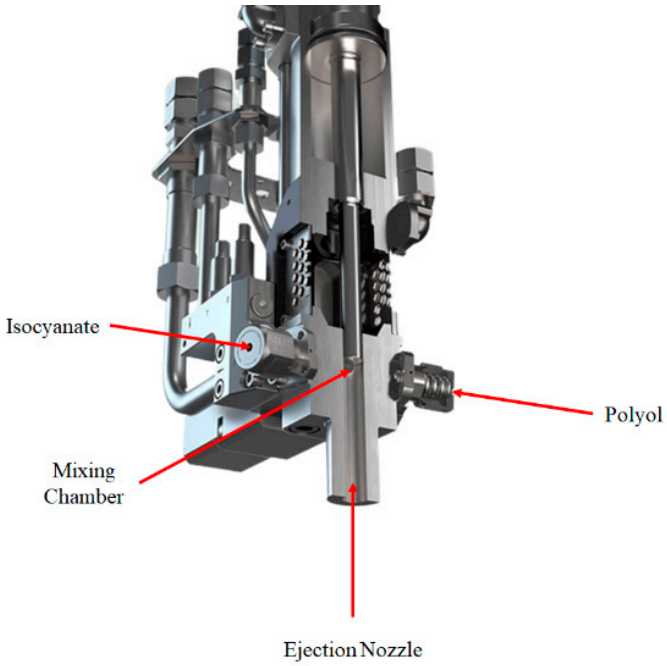


Figure 1. Configuration of foaming machine.

The foaming process involves the production of polyurethane foam constituting automobile seat cushion foam or automobile interior parts by mixing and foaming polyol and isocyanate. This process was performed within 1.11 s, and the mixing and foaming processes occurred when each raw material was foamed rather than generating and foaming a mixed solution. Therefore, information regarding the mixed raw materials is unknown. Thus, the process data for each raw material were collected by attaching flow, pressure, and temperature sensors to the part closest to the mixing head [24].

To measure the flow, pressure, and temperature of isocyanate and polyol, Kracht VC1 F4 PS, Keller PA-21Y, and PT 100 Ω type were used. In this study, the average value of the data measured for 1.11 s, which is the one-time foaming time, was calculated and used. The collected data were divided into two types: normal and defective. The classification was based on ±3% of the mass reference value, which was the same as the quality requirement of the finished-vehicle company. In the case of the mass of the produced polyurethane foam being higher than the reference value, the density of the tissue inside the cushion was high, resulting in poor ride comfort. However, if the mass of the produced polyurethane foam was lower than the reference value, the density of the tissue inside the cushion decreased, and the passenger's body was not well supported.

The mass of the final polyurethane foam product is related to the flow rate, temperature, and pressure. However, because each factor exhibits a nonlinear relationship with the others, this study considered these three factors; the labels are listed in Table 1. As various process data were collected, the dimensions of each factor differed. Furthermore, the algorithm being developed was not process-specific but general-purpose. To offset the characteristics of each factor, a normalization process was performed using the equation below, such that the maximum and minimum values of each factor were the same.

$$M = \frac{C - (C_{min} - W_n \times D)}{\{(C_{max} + W_n \times D) - (C_{min} - W_n \times D)\}} \quad (1)$$

where M is a normalized value, C is the data to be normalized, C_{max} and C_{min} are the maximum and minimum values of each factor, respectively, and D is the difference between C_{max} and C_{min}. This equation allows a value between 0 and 1 to be derived based on the maximum and minimum values for each factor. Here, normalization may be performed in a narrower range than 0–1 by adjusting the W_n factor. Based on previous research that has reported the possibility of deriving results with higher prediction performance and reliability by avoiding the normalization range of 0–1, this study employed a W_n value of 0.125 to limit the normalization range to a range from 0.1 to 0.9 [25]. The lower row in Table 1 shows the results of the normalization in Equation (1).

Table 1. Collected foaming process data.

	Isocyanate			Polyol			Label
	Flow	**Pressure**	**Temperature**	**Flow**	**Pressure**	**Temperature**	
Original Dataset	42.6	92.2	24.4	133.3	109.1	26.4	Fine
	42.1	93.4	24.4	133.3	109.7	26.3	Fine
	41.7	105	24.4	134.4	109.6	26.3	Fine
	42.1	100.9	24.8	135.4	109.6	26.3	Fine
	42.1	94.9	24.3	133.3	108.7	26.4	Defect
	41.2	90.7	24.	133.3	107.5	26.9	Defect
	41.2	109.5	24.9	133.3	108.6	26.5	Defect
	41.7	90	24.7	134.4	109.4	26.3	Defect
Normalized Dataset	0.616667	0.537258	0.493846	0.741975	0.338596	0.841935	Fine
	0.602778	0.55413	0.493846	0.741975	0.366667	0.835484	Fine
	0.591667	0.717223	0.493846	0.796296	0.361988	0.835484	Fine
	0.602778	0.659578	0.543077	0.845679	0.361988	0.835484	Fine
	0.602778	057522	0.481538	0.741975	0.319883	0.841935	Defect
	0.577778	0.516169	0.506154	0.796296	0.319883	0.848387	Defect
	0.577778	0.780492	0.555385	0.741975	0.315205	0.848387	Defect
	0.591667	0.506327	0.530769	0.796296	0.35632	0.835484	Defect

An overview of the data processing and defect detection methods is shown in Figure 2. The numbers of fine and defective data are 8474 and 187, respectively. Some labeled data are listed in Table 1. Tabular data were converted into image data. The tabular and image datasets were augmented using the synthetic minority oversampling technique (SMOTE) and cutout. Following preprocessing and augmentation, the tabular and image datasets were passed through an ANN and CNN for defect detection.

2.2. ANN

When addressing data imbalance problems using seat foam manufacturing process data, a challenge often arises in the case of an abundance of normal data, but a scarcity of defective data. This imbalance can result in biased and inaccurate machine learning models, particularly for minority labels. To address this issue, typically, oversampling techniques are employed, particularly to augment existing defective data. There are various methods for oversampling based on the nature of the data to be augmented. In this study, SMOTE was applied to augment the numerical data in the tabular datasets, and a cutout was utilized to augment the image data.

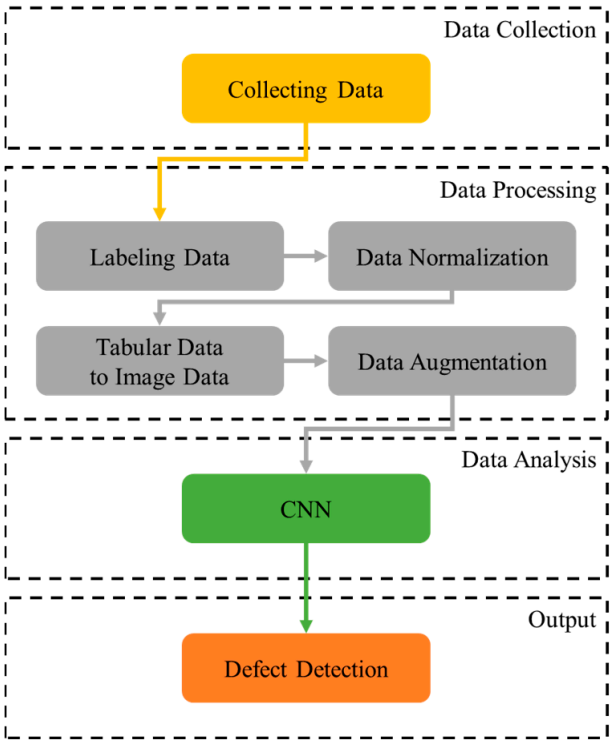


Figure 2. A flowchart representing the proposed defect detection method.

The synthetic minority oversampling SMOTE is used to address the issue of class imbalance in machine learning. Class imbalance occurs when certain classes in the dataset have significantly fewer instances than others, causing the model to be biased towards the majority class. SMOTE helps to mitigate this problem.

First, SMOTE begins by selecting data points from a minority class. Subsequently, it identifies the k-nearest neighbors for each selected data point and generates new data points by interpolating the selected data point and its neighbors. A random number between 0 and 1, denoted as λ, is chosen. The new data points are calculated as follows:

$$\text{Synthetic Data} = X + \lambda\,(X_{nn} - X) \tag{2}$$

Finally, the newly generated data points are added to the existing dataset. In this study, the values of k are chosen as 65, 98, and 130.

An ANN is a computational model inspired by biological neural networks in the human brain. ANNs typically comprise three main types of layers: input, hidden, and output. The input layer receives the initial data or features fed into the neural network. The number of neurons in the input layer corresponds to the number of input features. The neurons in the hidden layers perform computations on the input data. The network learns and extracts features from the input data in these hidden layers. Consequently, the output layer produces the final predictions or outputs of the neural network.

The ANN architecture in Figure 3 is simulated via Matlab 2022b on Windows 10. The model parameters for each layer and activation functions are listed in Table 2. In this process, the batch normalization layer and ReLU activation function were applied after the previous fully connected layer. This process was repeated three times to analyze the numerical data. Subsequently, to prevent overfitting, the model was passed through a dropout layer, and the results of the fully connected layer were output to the classification layer using the SoftMax activation function.

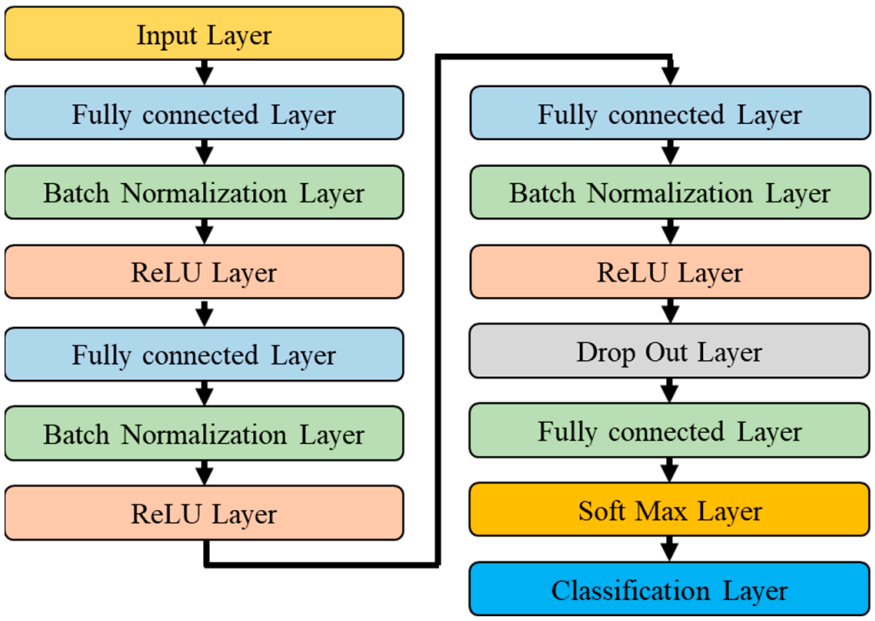


Figure 3. ANN model.

Table 2. ANN model parameters.

Layer	Feature Map	Learnables	Activation
Input Layer	6	-	
Fully Connected Layer	100	Weight 100 × 6, Bias 100 × 1	
Batch Normalization Layer	100	Offset 100 × 1, Scale 100 × 1	ReLU
Fully Connected Layer	50	Weight 50 × 100, Bias 50 × 1	
Batch Normalization Layer	50	Offset 50 × 1, Scale 50 × 1	ReLU
Fully Connected Layer	10	Weight 10 × 50, Bias 10 × 1	
Batch Normalization Layer	10	Offset 10 × 1, Scale 10 × 1	ReLU
Dropout Layer	10	-	
Fully Connected Layer	2	Weight 2 × 10, Bias 2 × 1	
Classification Layer	2	-	SoftMax

3. Proposed CNN Architecture

3.1. Data Creation and Augmentation for CNN

The process of changing the data from a normalized table to images was applied to a 2D CNN. In the image-creation phase, a 2 × 3 image was generated using six factors. Each image was labeled as fine or defective. Because alternative ordering results occur in diverse image forms, the order of the elements is crucial. To produce a coherent and significant visual depiction, we grouped the factors and exhibited factors collectively or in close proximity. Sample 2 × 3 images produced during image production are shown in Figure 4. The process of converting the data into an image was performed such that the normalized value could be composed of a grayscale value between 0 and 1. To specify a value for grayscale, the following was used: 0 means black, 1 means white. The determined gray color was then displayed in the generated image.

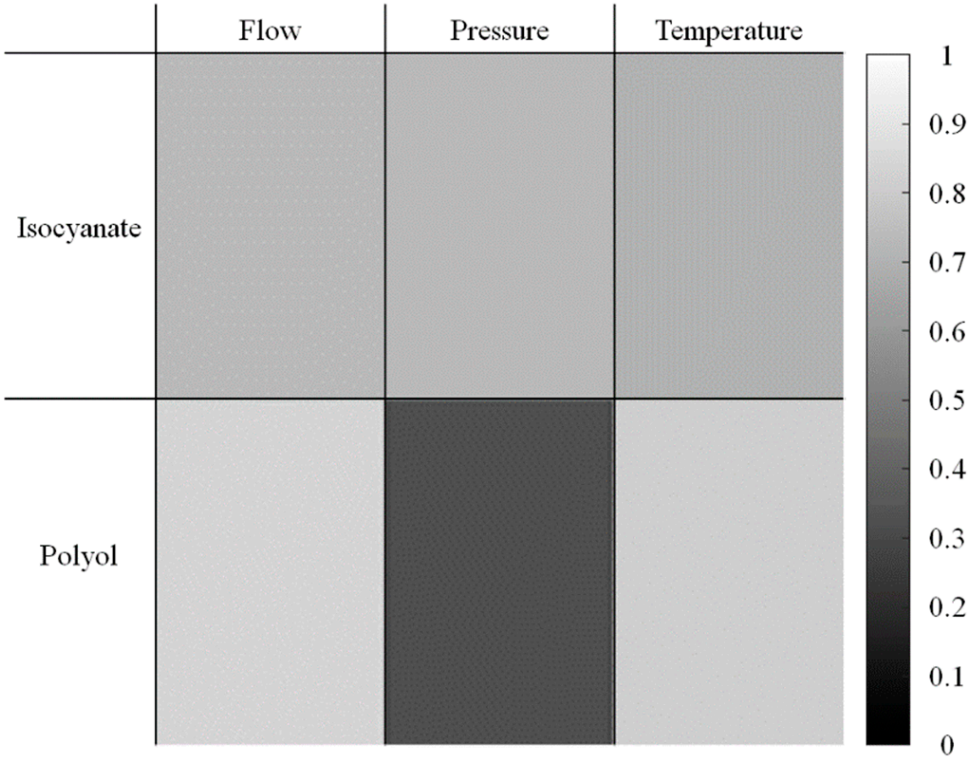
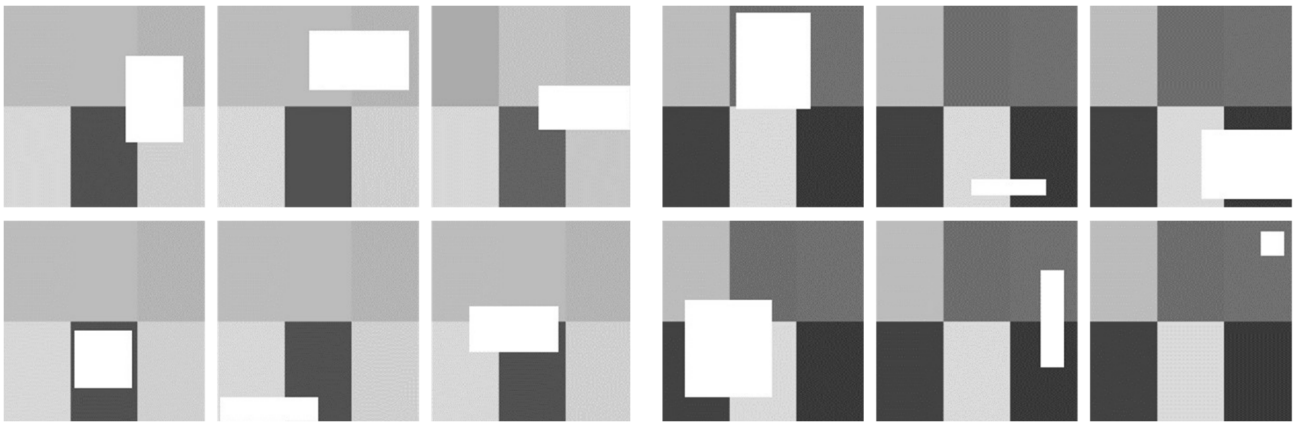


Figure 4. Image data creation from tabular data.

While collecting actual process data, there may be an imbalance in the number of normal and defective data points or a lack of collected data. Moreover, when learning is performed using such a dataset, an overfitting problem can occur with a high validation accuracy but low test accuracy. The simplest method to address this challenge involves increasing the amount of data artificially before proceeding with learning. This method is referred to as "data augmentation", and there exist methods to change the basic characteristics of an image, such as cropping a portion of the image, rotating the image, flipping the image symmetrically in the horizontal or vertical direction, and injecting noise in the image. Various studies have focused on methods for changing the geometrical features of an image, such as mix-up, which mixes two images into one, and random erase, which masks a part of the existing image with a random grayscale image [26]. The data augmentation method used in this study was eliminated, and the existing image was masked by a randomly sized white rectangle within the range of a minimum of 1×1 to a maximum of 112×112. When masked at the maximum size, 1/4 of the total image was covered. Furthermore, the masked position was randomly designated to prevent the problem of concentrated masking at a specific position [27].

In contrast to other methods, the image augmentation method using a white rectangular cutout is suitable for this study, where the classification results may vary depending on the grayscale values for each location in the image. Moreover, there is no change in or movement of normalized values for each location because the scale of an image does not change or the image is not augmented, such as in symmetric movement and rotation.

The augmentation results were used to generate the image shown in Figure 5. The figure (a) presents the result of augmenting the fine data, and (b) presents the result of augmenting the defect data. To solve the data shortage problem, the original image dataset is increased by factors of 2, 5, and 10.



(**a**) (**b**)

Figure 5. Augmented image data: (**a**) augmented fine data; (**b**) augmented defect data.

3.2. Two-Dimensional CNN Model

Through this process, sensor data from industrial sites, where image data could not be collected, were visualized and applied to the 2D CNN algorithm. The 2D CNN algorithm is a deep neural network algorithm conceived based on the working principle of the optic nerve in animals and is designed with a structure suitable for image data processing. A 2D CNN extracts the features of the data by reducing the size of the overall image and increasing the image dimensions as the image is input to the feature extraction layer and then passes through the sequentially constructed convolution and pooling layers. The extracted features are finally entered into the fully connected layer, and a final judgment is made. The algorithm used was based on CNN-WDI, which is a 2D CNN model proposed by Saqlain et al., and the number of nodes in the fully connected layer was changed to make it suitable for this study [15]. The structure of the 2D CNN model in Figure 6 is simulated through Matlab on Windows 10. The parameters such as the output size and activation function of each layer are listed in Table 3.

3.3. Optimization of Hyperparameters

CNNs contain several hyperparameters that must be carefully tuned to achieve an optimal performance in various tasks. Certain key hyperparameters in CNNs include the minibatch size, max epoch, learning rate, and dropout. The minibatch size is the number of data samples used in one training iteration. This balances efficiency and accuracy, thereby affecting the convergence speed. The max epoch signifies the maximum number of training cycles through the dataset, which is crucial for model learning to avoid underfitting or overfitting. The learning rate dictates the step size for updating the neural network weights during training. This influences the convergence and model accuracy, which require careful tuning. Dropout is a regularization method that prevents overfitting. It randomly drops neurons, thereby enhancing model generalization. The dropout rate controls the fraction of dropped neurons, which is crucial for robustness.

The selection of hyperparameter values significantly affects the performance of machine learning models. In this study, the grid search method, which is a systematic approach for determining the best combination of hyperparameters by evaluating all possible combinations within a specified range for each hyperparameter, was used. The ranges of the hyperparameters are listed in Table 4. Subsequently, a grid containing all possible combinations of the specified hyperparameters was generated. For each combination in the grid, the model was trained on the training data, and its performance was evaluated using cross-validation techniques (such as k-fold cross-validation). This provided an unbiased estimate of model performance for each set of hyperparameters. Thus, the combination of hyperparameters resulted in the best performance based on the loss function. This combination represents the optimal set of hyperparameters for the proposed model. Once the

optimal hyperparameters were determined, the final model was trained using these values for the entire training dataset. The range of hyperparameters was determined through experiments and theories and by randomly substituting values within the range [28]. To reduce the computational time required for iterative operations during this process, the size of the images was reduced to 25 × 25 × 3. Table 5 provides the optimized values of the hyperparameters used in the optimization method. After the hyperparameters were determined, the model was trained with image data of the original size (224 × 224 × 3). The training time for the CNN model was approximately 4 h.

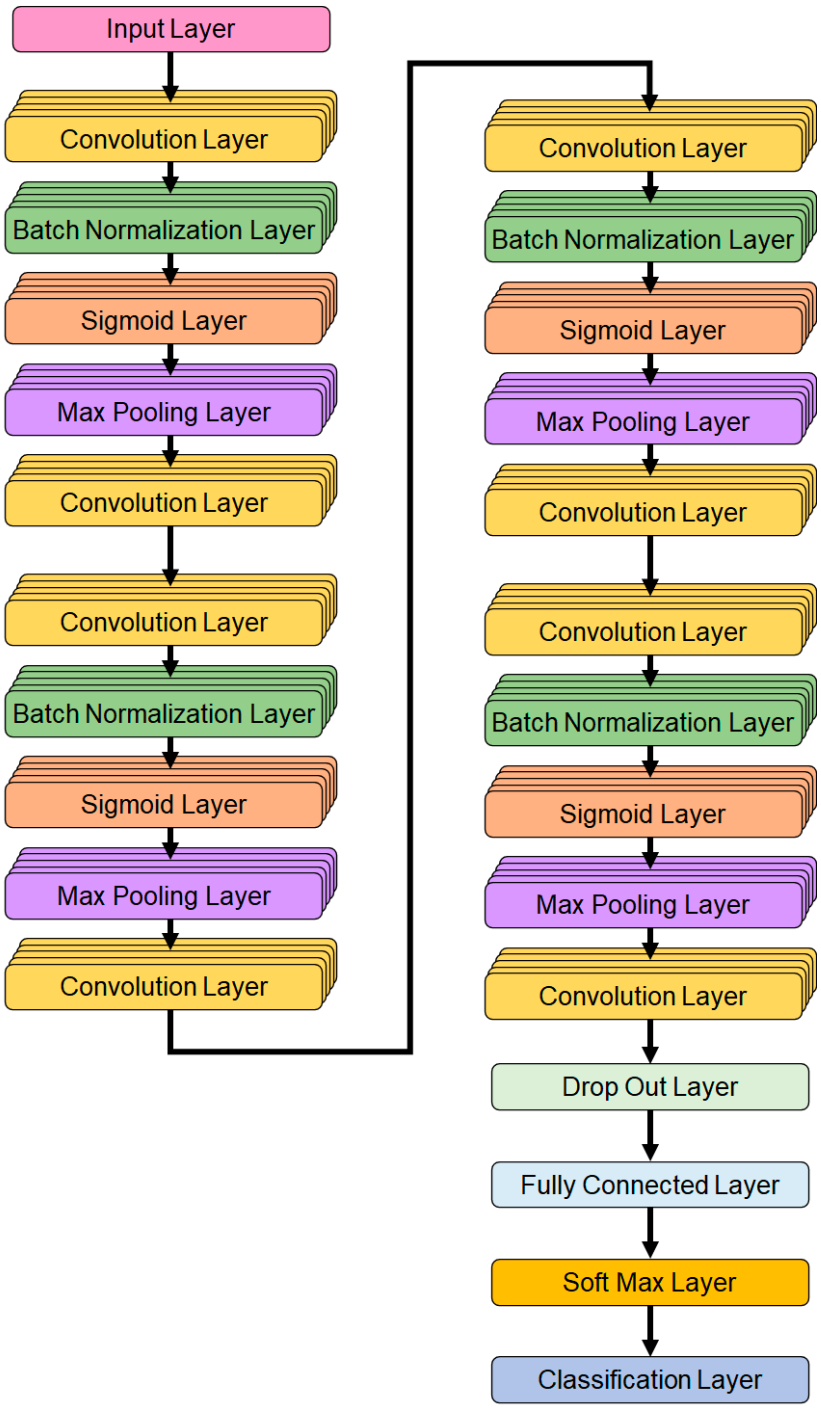


Figure 6. CNN-WDI model.

Table 3. CNN model parameters.

Layer	Activation	Learnables	Activation
Image Input Layer	224 × 224 × 3	-	
Convolution Layer	224 × 224 × 16	Weight 3 × 3 × 3 × 16, Bias 1 × 1 × 16	
Batch Normalization Layer	224 × 224 × 16	Offset 1 × 1 × 16, Scale 1 × 1 × 16	Sigmoid
Max Pooling Layer	112 × 112 × 16	-	
Convolution Layer	112 × 112 × 16	Weight 3 × 3 × 16 × 16, Bias 1 × 1 × 16	
Convolution Layer	112 × 112 × 32	Weight 3 × 3 × 16 × 32, Bias 1 × 1 × 32	
Batch Normalization Layer	112 × 112 × 32	Offset 1 × 1 × 32, Scale 1 × 1 × 32	Sigmoid
Max Pooling Layer	56 × 56 × 32	-	
Convolution Layer	56 × 56 × 32	Weight 3 × 3 × 32 × 32, Bias 1 × 1 × 32	
Convolution Layer	56 × 56 × 64	Weight 3 × 3 × 32 × 64, Bias 1 × 1 × 64	
Batch Normalization Layer	56 × 56 × 64	Offset 1 × 1 × 64, Scale 1 × 1 × 64	Sigmoid
Max Pooling Layer	28 × 28 × 64	-	
Convolution Layer	28 × 28 × 64	Weight 3 × 3 × 64 × 64, Bias 1 × 1 × 64	
Convolution Layer	28 × 28 × 128	Weight 3 × 3 × 64 × 128, Bias 1 × 1 × 128	
Batch Normalization Layer	28 × 28 × 128	Offset 1 × 1 × 128, Scale 1 × 1 × 128	Sigmoid
Max Pooling Layer	14 × 14 × 128	-	
Convolution Layer	14 × 14 × 128	Weight 3 × 3 × 128 × 128, Bias 1 × 1 × 128	
Dropout	14 × 14 × 128	-	
Fully Connected Layer	1 × 1 × 2	-	
Classification Layer	1 × 1 × 2	-	SoftMax

Table 4. Range of grid search and step.

Hyperparameter	Range	Step
Initial Learning Rate	0.001~0.01	0.05
Max Epoch	10~50	10
Minibatch Size	10~300	50
Dropout Rate	0~0.3	0.1

Table 5. Optimized values of the hyperparameters.

Algorithm	Dataset	Optimization Hyperparameter			
		Minibatch	Max Epoch	Dropout	Learning Rate
ANN	k-65 SMOTE	100	50	0	0.005
	k-98 SMOTE	300	50	0	0.001
	k-130 SMOTE	150	50	0	0.005
CNN	2× Augmentation	100	50	0.2	0.001
	5× Augmentation	150	30	0.1	0.001
	10× Augmentation	100	20	0.3	0.001

4. Results and Discussion

To derive the test results of the learned model, a confusion matrix was configured, where *Tp*, *Fp*, *Fn*, and *Tn* represent the numbers of data points classified as true positive, false positive, false negative, and true negative, respectively.

It was evaluated using various performance measures, such as accuracy, precision, recall, and *F*1 score [29]. The *Accuracy* of a classifier defines the frequency with which it accurately predicts over the entire dataset, and is defined as follows:

$$Accuracy = \frac{Tp + Tn}{Tp + Tn + Fp + Fn} \tag{3}$$

Precision and *Recall* indicate that both performance measures are inversely proportional to each other, and each has various classification-measuring qualities.

$$Precision = \frac{Tp}{Tp + Fp} \tag{4}$$

$$Recall = \frac{Tp}{Tp + Fn} \tag{5}$$

The *F*1 score calculates the harmonic mean of precision and recall and is defined as follows: The *F*1 score is an interpretation of actual and predicted probabilities. If these probabilities are close to each other, then the *F*1 score exhibits a higher result, and vice versa.

$$F1\ score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{6}$$

To evaluate the execution time of defect detection, we used the GPU Nvidia GeForce RTX 3070 and CPU Intel i9 10,900, resulting in an execution time of 0.089 s. The performance evaluation metrics are listed in Table 6. Applying the oversampled data to the ANN, the dataset with the k value set to 130 exhibited the highest accuracy of 84.82% and a fallout value of 0.2857. In the CNN utilizing cutout, the image data were augmented tenfold, resulting in an accuracy of 97.32% and a fallout value of 0.0357. Consequently, a performance comparison between an ANN applied to tabular data and a CNN utilizing tabular data converted into image data revealed the superior predictive accuracy and fallout values of the CNN.

Table 6. Performance evaluation index by dataset.

Algorithm	Dataset	Performance Index				
		Accuracy	**Precision**	**Recall**	**F1 Score**	**Fallout**
ANN	k-65 SMOTE	0.8304	0.7606	0.9643	0.8540	0.3036
	k-98 SMOTE	0.8393	0.7639	0.9821	0.8594	0.3036
	k-130 SMOTE	0.8482	0.7746	0.9821	0.8661	0.2857
CNN	2× Augmentation	0.9554	0.9474	0.9643	0.9558	0.0526
	5× Augmentation	0.9643	0.943	0.9643	0.9643	0.0357
	10× Augmentation	0.9732	0.9649	0.9821	0.9735	0.0357

This confirmed that the tabular data applied to the ANN were effectively transformed into grayscale images. Moreover, the spatial context information of these image files formed an efficient array for extracting the features of the seat-forming process. Employing convolution layers for feature extraction and max pooling layers for downsampling the image allowed the spatial hierarchical structure of the image data to be learned well. Consequently, the preservation of specific parts of the image and assistance in feature extraction were evident. The CNN model used in this study repeated the structure of the convolution layer–batch normalization–sigmoid–max pooling–convolution layer four times. In addition, previous studies have reported that accuracy escalates with each repetition, indicating effective feature extraction [30].

5. Conclusions

This study discussed the effectiveness of using deep learning, specifically 2D CNN algorithms, to predict defects in manufacturing processes with nonlinear characteristics. The focus was on predicting defects in seat foaming manufacturing using both ANNs and CNNs. To address data imbalances, oversampling techniques were employed, including SMOTE for the ANN and cutout augmentation for the CNN. The CNN outperformed the ANN in terms of prediction accuracy, particularly with the 10-fold augmented dataset using cutout, achieving 97.32% accuracy and a fallout value of 0.0357. Moreover, the importance of preprocessing techniques such as grayscale conversion for image data was emphasized and it was concluded that a CNN, with its ability to extract spatial features and repeated convolution layers, was more effective than an ANN in predicting defects using transformed image data in manufacturing processes.

Regarding the reliability of prediction or process analysis, it is effective to predict defects in processes with nonlinear characteristics using deep learning. In particular, utilizing a 2D CNN algorithm with excellent classification performance is effective, and data in image form are necessary to apply the 2D CNN algorithm. However, extracting distinct features, particularly when mixing raw materials, is challenging, and the process of injecting them into a mold is limited by the collection and analysis of image data. This process requires the conversion of numerical data into images. In this study, the concept of each factor's range or unit was eliminated through the normalization of the numerical data, thereby facilitating classification based on its location within the entire range. It is believed that these normalization results can be converted into grayscale values to allow a model to learn a range of converted grayscale values instead of specific numbers, thereby enhancing the classification performance. As a part of future work, we aim to explore and implement various methods to reduce the training time of the 2D CNN model. This will include investigating the impact of optimizing the size of the image dataset, among other potential strategies.

Author Contributions: J.-S.O. takes the primary responsibility for this research as the principal investigator and drafted the manuscript. N.-H.C. contributed to the analysis, algorithm, and writing of the manuscript. J.W.S. contributed to the investigation and writing—review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: This results was partially supported by the "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) (2021RIS-004) and the Korea government (MSIT) (No. 2022R1F1A1074691).

Data Availability Statement: The data used to support the findings of this study are included within the article.

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

References

1. Soualhi, M.; Nguyen, K.T.P.; Medjaher, K. Pattern recognition method of fault diagnostics based on a new health indicator for smart manufacturing. *Mech. Syst. Signal Process.* **2020**, *142*, 106680. [CrossRef]
2. Booyse, W.; Wilke, D.N.; Heyns, S. Deep digital twins for detection, diagnostics and prognostics. *Mech. Syst. Signal Process.* **2020**, *140*, 106612. [CrossRef]
3. Jang, I.; Bae, G.; Kim, H. Metal forming defect detection method based on recurrence quantification analysis of time-series load signal measured by real-time monitoring system with bolt-type piezoelectric sensor. *Mech. Syst. Signal Process.* **2022**, *180*, 109457. [CrossRef]
4. Chen, Y.; Rao, M.; Feng, K.; Zuo, M.J. Physics-Informed LSTM Hyperparameters Selection for Gearbox Fault Detection. *Mech. Syst. Signal Process.* **2022**, *171*, 108907. [CrossRef]
5. Han, T.; Xie, W.; Pei, Z. Semi-Supervised Adversarial Discriminative Learning Approach for Intelligent Fault Diagnosis of Wind Turbine. *Inf. Sci.* **2023**, *648*, 119496. [CrossRef]
6. Chen, Y.; Rao, M.; Feng, K.; Niu, G. Modified Varying Index Coefficient Autoregression Model for Representation of the Nonstationary Vibration from a Planetary Gearbox. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–12. [CrossRef]
7. Yao, J.; Han, T. Data-Driven Lithium-Ion Batteries Capacity Estimation Based on Deep Transfer Learning Using Partial Segment of Charging/Discharging Data. *Energy* **2023**, *271*, 127033. [CrossRef]

8. Im, S.J.; Oh, J.S.; Kim, G.-W. Simultaneous Estimation of Unknown Road Roughness Input and Tire Normal Forces Based on a Long Short-Term Memory Model. *IEEE Access* **2022**, *10*, 16655–16669. [CrossRef]
9. Chien, J.C.; Wu, M.T.; Lee, J.D. Inspection and classification of semiconductor wafer surface defects using CNN deep learning networks. *Appl. Sci.* **2020**, *10*, 5340. [CrossRef]
10. Sahu, C.K.; Young, C.; Rai, R. Artificial intelligence (AI) in augmented reality (AR)-assisted manufacturing applications: A review. *Int. J. Prod. Res.* **2021**, *59*, 4903–4959. [CrossRef]
11. Schmidhuber, J. Deep learning in neural networks: An overview. *Neural Netw.* **2015**, *61*, 85–117. [CrossRef] [PubMed]
12. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
13. Glaeser, A.; Selvaraj, V.; Lee, S.; Hwang, Y.; Lee, K.; Lee, N.; Lee, S.; Min, S.; Min, S. Applications of deep learning for fault detection in industrial cold forging. *Int. J. Prod. Res.* **2021**, *59*, 4826–4835. [CrossRef]
14. Nakazawa, T.; Kulkarni, D.V. Wafer map defect pattern classification and image retrieval using convolutional neural network. *IEEE Trans. Semicond. Manuf.* **2018**, *31*, 309–314. [CrossRef]
15. Saqlain, M.; Abbas, Q.; Lee, J.Y. A deep convolutional neural network for wafer defect identification on an imbalanced dataset in semiconductor manufacturing processes. *IEEE Trans. Semicond. Manuf.* **2020**, *33*, 436–444. [CrossRef]
16. Liu, J.; Shao, Y. Overview of dynamic modelling and analysis of rolling element bearings with localized and distributed faults. *Nonlinear Dyn.* **2018**, *93*, 1765–1798. [CrossRef]
17. Wang, L.-H.; Zhao, X.-P.; Wu, J.-X.; Xie, Y.-Y.; Zhang, Y.-H. Motor fault diagnosis based on short-time Fourier transform and convolutional neural network. *Chin. J. Mech. Eng.* **2017**, *30*, 1357–1368. [CrossRef]
18. Chen, K.; Zhou, X.-C.; Fang, J.-Q.; Zheng, P.-F.; Wang, J. Fault feature extraction and diagnosis of gearbox based on EEMD and deep briefs network. *Int. J. Rotating Mach.* **2017**, *2017*, 9602650. [CrossRef]
19. Guo, S.; Yang, T.; Gao, W.; Zhang, C. A novel fault diagnosis method for rotating machinery based on a convolutional neural network. *Sensors* **2018**, *18*, 1429. [CrossRef]
20. Sun, W.F.; Yao, B.; Zeng, N.; Chen, B.; He, Y.; Cao, X.; He, W. An intelligent gear fault diagnosis methodology using a complex wavelet enhanced convolutional neural network. *Materials* **2017**, *10*, 790. [CrossRef]
21. Zhao, M.; Kang, M.; Tang, B.; Pecht, M. Deep residual networks with dynamically weighted wavelet coefficients for fault diagnosis of planetary gearboxes. *IEEE Trans. Ind. Electron.* **2018**, *65*, 4290–4300. [CrossRef]
22. Sezer, O.B.; Ozbayoglu, A.M. Algorithmic financial trading with deep convolutional neural networks: Time series to image conversion approach. *Appl. Soft Comput.* **2018**, *70*, 525–538. [CrossRef]
23. Lee, E.; Nam, M.; Lee, H. Tab2vox: CNN-based multivariate multilevel demand forecasting framework by tabular-to-voxel image conversion. *Sustainability* **2022**, *14*, 11745. [CrossRef]
24. Choi, N.K. Defect Prediction Model of Automobile Interior Parts in Manufacturing Process. Master Thesis, Kongju National University, Cheonan, Choongnam, Republic of Korea, 2023.
25. Adnan, R.; Ruslan, F.A.; Samad, A.M.; Zain, Z.M. New Artificial Neural Network and Extended Kalman Filter Hybrid Model of Flood Prediction System. In Proceedings of the 2013 IEEE 9th International Colloquium on Signal Processing and its Applications, Kuala Lumpur, Malaysia, 8–10 March 2013; pp. 252–257. [CrossRef]
26. Shorten, C.; Khoshgoftaar, T.M. A survey on image data augmentation for deep learning. *J. Big Data* **2019**, *6*, 1–48. [CrossRef]
27. DeVries, T.; Taylor, G.W. Improved Regularization of Convolutional Neural Networks with Cutout. *arXiv* **2017**, arXiv:1708.04552.
28. Restrepo Rodríguez, A.O.; Casas Mateus, D.E.; Gaona García, P.A.; Montenegro Marín, C.E.; González Crespo, R. Hyperparameter optimization for image recognition over an ar-sandbox based on convolutional neural networks applying a previous phase of segmentation by color–space. *Symmetry* **2018**, *10*, 743. [CrossRef]
29. Sokolova, M.; Japkowicz, N.; Szpakowicz, S. *Beyond Accuracy, F-Score and ROC: A Family of Discriminant Measures for Performance Evaluation*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1015–1021.
30. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.