

# Development of a CNN-based real-time monitoring algorithm for additively manufactured molybdenum

Eun-Su Kim<sup>a</sup>, Dong-Hee Lee<sup>b,\*</sup>, Gi-Jeong Seo<sup>c</sup>, Duck-Bong Kim<sup>c</sup>, Seung-Jun Shin<sup>d</sup>

<sup>a</sup> Department of NAND SE, SK hynix, 115 Jikji-daero, 307 beon-gil, Heungdeok-gu, Cheongju 28435, Republic of Korea

<sup>b</sup> Department of Industrial Engineering, Sungkyunkwan University, 2066 Seobu-ro, Jangnan-gu, Suwon 16419, Republic of Korea

<sup>c</sup> Department of Manufacturing and Engineering Technology, Tennessee Technological University, 1 William L Jones Drive, Cookeville, TN 38505, United States

<sup>d</sup> Division of Interdisciplinary Industrial Studies, Hanyang University, 222 Wangsimni-ro, Seongdong-gu, Seoul 04763, Republic of Korea

## ARTICLE INFO

### Keywords:

Real-time monitoring  
Molybdenum  
Wire + arc additive manufacturing  
Process signatures  
Convolutional neural network

## ABSTRACT

A convolutional neural network (CNN)-based real-time monitoring algorithm is present to detect an abnormal wire + arc additive manufacturing (WAAM) process for molybdenum. The proposed algorithm consists of three modules: image conversion, CNN prediction, and real-time monitoring. The image conversion module changes the form of a time-series voltage waveform data into voltage image data. The CNN prediction module classifies each voltage image into a normal or abnormal image. The real-time monitoring module expresses the results of the CNN prediction model on a real-time dashboard. Experiments for single beads of molybdenum materials were performed to validate the performance of the proposed algorithm. It was observed that abnormal WAAM processes are detected in real-time with high accuracy. In addition, a sensitivity analysis with respect to different intervals and bandwidths of the voltage image data was conducted, which are the main input parameters of the proposed method. Based on this investigation, guidelines for setting the interval and bandwidth were established. Finally, the effectiveness of the CNN classifiers was validated by applying a class-activation mapping method. It was concluded that the CNN classifiers were adequately trained because they captured the critical regions in the voltage images for both normal and abnormal cases.

## 1. Introduction

Molybdenum (Mo) is a refractory metal with good thermal property, creep resistance, wear resistance, and thermal expansion and is used in furnaces, semiconductors, and nuclear power applications that have specific material property requirements (e.g., high temperature) [1]. However, Mo is a hard-to-machine material owing to its high melting point and hardness. In addition, it is an expensive material, so subtractive processes are not efficient options, when a part with a high buy-to-fly ratio is machined. In contrast, additive manufacturing (AM) is considered as a potential solution to fabricate highly complex and hard-to-machine Mo structures, saving time and cost. Among metal AM processes, a wire and arc additive manufacturing (WAAM) process can fabricate large-scale metal structures owing to the inexpensive system setup cost, high deposition rates, and energy efficiency [2]. In particular, the inexpensive cost and energy efficiency can provide a capability to deposit refractory alloy structures (e.g., tungsten and Mo) [3,4].

One of difficulties in applying AM is that quality control is largely limited to offline techniques in the absence of real-time process monitoring, resulting in high scrap rates [5,6]. This is more evident when the costs of materials such as Mo are high. Recently, artificial intelligence (AI) has drastically advanced monitoring and control in many areas. Therefore, AM researchers increasingly apply advanced AI to AM, particularly real-time monitoring and control [7]. However, three difficulties remain: (1) the lack of data owing to difficulties in measurements in metal AM [8], (2) difficulty in obtaining accurate and reliable data, and (3) difficulty in developing accurate real-time processing models.

To cope with these difficulties, methods that can be accurately processed in real-time using small datasets should be developed. The data collected for real-time monitoring during the metal AM process can be divided into two types: 1) one-dimensional (1D) data, 2D, and 3D data. 1D data include digital signals, such as the power, current, wire feed rate, temperature, and vibration collected by sensors [9]. They are

\* Corresponding author.

E-mail address: [dhee@skku.edu](mailto:dhee@skku.edu) (D.-H. Lee).

<sup>1</sup> ORCID: 0000-0001-8549-8992

primarily time-series data types in which values are collected over time. Because of this property, methods such as long short-term memory (LSTM), which is a type of recurrent neural network (RNN), or an autoencoder, which is a type of neural network, are primarily used. However, they are unsupervised learning methods, which have the disadvantage of decreasing accuracy with small datasets.

2D data are images or video frames collected by cameras. They are advantageous in that they clearly show whether the AM process is normal or abnormal; thus, a training dataset can be prepared. Based on the training dataset, for example, convolutional neural networks (CNN) model, which shows good classification performances in image classification can be used. However, it is often difficult to obtain a high quality of 2D data because metal AM uses high temperatures, special cameras of high cost are required, and several conditions for video recording such camera location and installation angle, are significantly affected the quality of the video frames [10]. 3D data are profiles of surface or internal scans that are rendered in the form of a 3D file. They also require special equipment, such as a computer tomography (CT) system or profilometer, making real-time applications more challenging [11].

In this study, we developed a real-time monitoring algorithm for WAAM'ed Mo using a CNN model. The proposed real-time monitoring algorithm overcomes the drawbacks of existing monitoring systems using both 1D and 2D data. The existing monitoring approaches using 1D data can be easily implemented because collecting 1D data is easy; however, the monitoring based on 1D data often shows low accuracy. Although the ones using 2D data have high accuracy, they are challenging to implement due to the harsh recording conditions resulted from high temperature of the AM process. The proposed method detects process abnormality based on a particular 1D data, voltage data. Based on certain bandwidths, voltage data is converted into voltage 'image' data. After converting the voltage data to voltage image data, we labeled the voltage image data by matching them with 2D data. Once the CNN model is trained based on the labeled data, it detects the abnormality in the WAAM process whenever 1D data is collected in real-time. We validated the CNN model in terms of accuracy and achieved an accuracy of 96%. The proposed method can be easily implemented in the WAAM process because it works with 1D data, and it can detect the abnormality of the WAAM process for Mo in real-time with high accuracy by using 2D data for training the CNN model.

The remainder of this paper is organized as follows. In Section 2, existing methods related to real-time monitoring for AM, including the conversion from 1D data to voltage image data, are reviewed. The proposed algorithm, experiment, and sensitivity analysis are presented in Section 3. Finally, conclusions are presented in Section 4.

## 2. Literature review

### 2.1. Wire + arc additive manufacturing (WAAM)

WAAM has the potential for significant industrial impact owing to its ability to create large metal components with a high deposition rate, low equipment cost, high material utilization, and environmental friendliness (Wu et al., 2018). WAAM can be categorized into three techniques according to the nature of the heat source: (1) gas metal arc welding (GMAW)-based WAAM, (2) gas tungsten arc welding (GTAW)-based WAAM, and (3) plasma arc welding (PAW)-based WAAM.

GMAW-based WAAM is a welding process in which an electric arc forms between a consumable wire electrode and the workpiece metal [12]. Excessive heat input, metal spatter, and arc wandering are three of its unavoidable problems, adversely affecting the geometrical accuracy and quality of the deposited parts [13]. However, the deposition rate is 2–3 times higher than that of the other two methods. GTAW-based WAAM uses a non-consumable tungsten electrode with a separately fed wire to produce the weld deposit. This method can effectively avoid arc wander, and metal spatter and decrease the heat accumulation in the GMAW-based WAAM deposition process. However, the GTAW

deposition rate is lower than that of the GMAW-based WAAM. Thus, if a high deposition rate is prioritized, GMAW may be preferred over GTAW, although its quality and process stability are generally lower [14,15]. In PAW-based WAAM, the arc energy density can be thrice that of GTAW, causing less weld distortion and smaller welds with higher welding speeds [16]. Although PAW provides the highest energy density electrical arc that enables high travel speeds and high-quality welds with minimized distortion, it typically requires the most extensive capital expenditure [17].

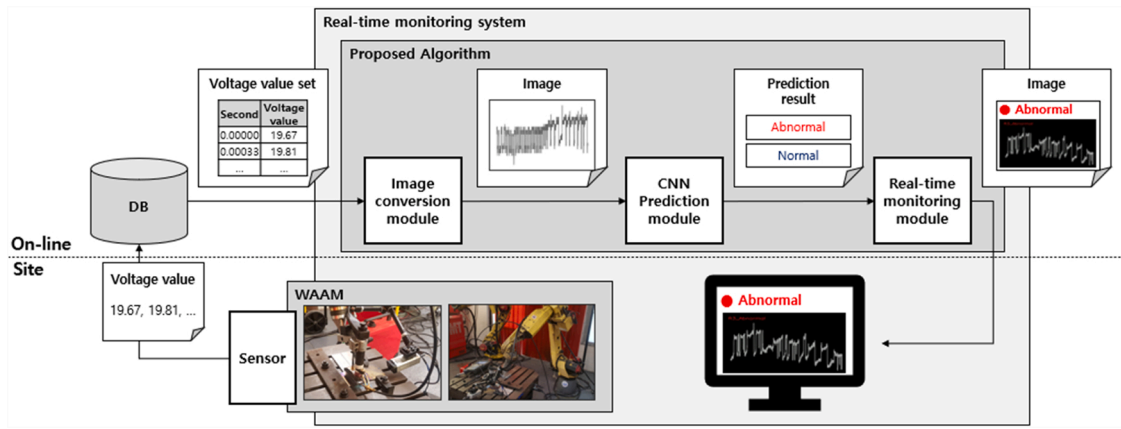
### 2.2. Real-time monitoring for AM

In this section, real-time monitoring methods for AM are reviewed based on 1D, 2D, and 3D data. First, 3D data-based monitoring methods are reviewed. In Wang et al. [11], Mo rods were fabricated using the GTAW-based WAAM process, and internal defects were detected by scanning them with a CT system. Through statistical data analysis, they demonstrated three main types of defects in Mo: (1) small spherical pores, (2) inverted pear-shaped pores, and (3) cavities. Li et al. [18] proposed a deep-learning-based quality identification method for metal AM processes. They used a publicly available real-world high-resolution image dataset from laser-based powder bed fusion processes using a stereomicroscope. They demonstrated a semisupervised identification consistency-based method that was robust against poor data quality. These 3D data-based monitoring methods showed good detection capability; however, collecting CT or stereomicroscope data is challenging in practice because due to high cost and long time of collection. In addition, these approaches cannot provide real-time control, and defect detection is delayed, causing a waste of time and material. This is more evident when the cost of materials is high, and the product is large. The following studies were conducted using 1D or 2D data collected during the process in real-time to apply real-time monitoring to solve such problems.

Grasso et al. [19] proposed statistical monitoring for the metal process using a support vector data description. Ren et al. [20] demonstrated an in-situ monitoring and signal processing method for laser metal AM by combining an unsupervised deep learning technique and emission spectroscopy. Guo et al. [21] proposed an in-situ monitoring system for the WAAM process with a hollow electromagnetic acoustic transducer that could adapt to a high-temperature and high-vacuum environment. These real-time monitoring systems using 1D data have efficient acquisition, storage, and analysis costs. However, compared to 2D data, such as video frames, it is difficult to determine the meaning of the information, the features are difficult to extract, and it is challenging to determine whether the process is abnormal. This results in low accuracy for detecting abnormality in AM process. In contrast, labeling 2D data, such as images or video frames, is relatively more straightforward. Therefore, real-time monitoring systems using 2D data frequently implement a CNN, facilitating supervised learning with proven efficiency and accuracy. Focusing on these advantages, the following studies were conducted.

Yang et al. [22] proposed a CNN-based real-time melt-pool classification method for AM. The data used were melt-pool images captured from a laser melting powder fusion build using a high-speed camera. Their model classification accuracy was approximately 91%. Bacioiu et al. [23] designed a system for assessing the quality of tungsten inert gas welding with the potential for application in real-time using a CNN, achieving an accuracy of 93.4%. The data they used was high dynamic range images representing the weld pool in visible spectra balance by offsetting the intense arc light. Caggiano et al. [24] proposed an online defect recognition platform using a bi-stream deep CNN, and its accuracy was as high as 99.4% for selective laser melting. The data they used was processed images extracted layer-by-layer from videos of the entire selective laser melting test.

Most of the existing CNN-based monitoring methods follow common two steps: Step 1) CNN model training & validation, Step 2) process



**Fig. 1.** The image conversion module collects the voltage values and converts them into waveform images in real-time. Whenever the converted image is generated, the CNN prediction module determines whether the WAAM process is normal or abnormal by classifying the converted image. Finally, the real-time monitoring module transfers the converted images with their classification results to a dashboard. Users can then recognize whether the WAAM process is normal or abnormal in real-time by observing the dashboard.

monitoring using the trained CNN model. In Step 2, the trained CNN model judges if the process is abnormal by classifying 2D data such as video frames that are mostly collected in a real-time basis. That is, the trained CNN models work based on 2D data in Step 2. However, the acquisition, storage, and analysis costs of 2D data, particularly high-temperature metal AM data, are high. Additionally, the 2D data can be easily affected or differ with respect to the measurement environment. For example, images can be affected by different viewing angles or light intensities; they may contain noises, such as fumes and spatter, which makes acquiring accurate images difficult. For these reasons, the existing CNN-based monitoring method is often difficult to use in real practice.

The proposed method suggests a hybrid of 1D and 2D data-based methods to overcome the aforementioned difficulties. The basic concept of the proposed method is to convert voltage 1D data into voltage image data, which are 2D data, and monitor the WAAM process by analyzing converted voltage image data. Thus, we can combine the easiness of collecting 1D data and the accuracy of learning a classification model from 2D data. Both proposed and existing methods requires 2D data in Step 1. However, the proposed method works based on 1D data in Step 2 unlike the existing methods. This is the main advantage of the proposed method over the existing method. On the surface, it seems that the proposed method also works based on 2D data since the CNN classifier is used. However, the 2D images used in the proposed method is a simple representation of 1D voltage time-series data; thus, it can be easily obtained. This makes the proposed method more practical and be easy to use in practice. The following section explains the details.

### 3. Proposed algorithm

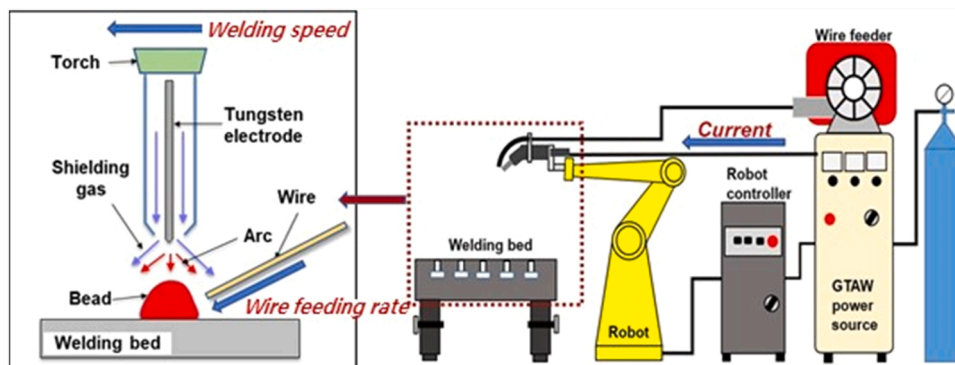
The proposed algorithm aims to build a monitoring system to detect whether the WAAM process is normal or abnormal in real-time. It collects voltage data from the WAAM process and conducts a real-time detection system based on that. Fig. 1 shows three modules of the algorithm: image conversion, CNN prediction, and real-time monitoring modules.

In Section 3.1, the experimental designs of the GTAW-based WAAM process are explained. Additionally, various data types, such as process signature, video, and image data collected from the experiments, are presented. In Section 3.2, the implementation of the three modules is described. In Section 3.3, the real-time monitoring system using the three modules is explained.

#### 3.1. Problem definition

The previously developed GTAW-based WAAM process was utilized to collect the data required to implement the proposed algorithm [25]. According to experimental conditions, the experiments were conducted as “bead on plate tests,” which generated Mo beads on a plate.

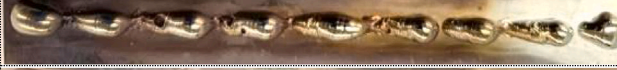
The experimental conditions were determined based on three WAAM process variables: current, welding speed, and feeding rate. Fig. 2 shows three process variables and the schematic of the WAAM system. Each variable had three levels of current: 250, 275, and 300 A; welding speed: 250, 275, and 300 cm/min; and feeding rate: 144, 180, and 216 cm/min. There were three variables, and each variable had three levels; theoretically, a total of 27 ( $=3^3$ ) experiments were possible. However,



**Fig. 2.** The WAAM system consists of 6-axis robot arm with GTAW power source and wire feeder. the three process variables are denoted by red color.



**Table 1**  
Experimental conditions.

| Test number | Current (A) | Welding Speed (CPM) | Wire feed speed (CPM) | Bead image                                                                           |
|-------------|-------------|---------------------|-----------------------|--------------------------------------------------------------------------------------|
| DOE-1       | 250         | 20                  | 180                   |    |
| DOE-2       | 300         | 20                  | 180                   |    |
| DOE-3       | 250         | 40                  | 180                   |    |
| DOE-4       | 300         | 40                  | 180                   |    |
| DOE-5       | 250         | 30                  | 180                   |    |
| DOE-6       | 300         | 30                  | 144                   |    |
| DOE-7       | 250         | 30                  | 216                   |    |
| DOE-8       | 300         | 30                  | 216                   |    |
| DOE-9       | 275         | 20                  | 144                   |    |
| DOE-10      | 275         | 40                  | 144                   |   |
| DOE-11      | 275         | 20                  | 216                   |  |
| DOE-12      | 250         | 30                  | 180                   |  |
| DOE-13      | 300         | 40                  | 216                   |  |
| DOE-14      | 275         | 30                  | 180                   |  |
| DOE-15      | 250         | 20                  | 144                   |  |
| Random-1    | 275         | 20                  | 200                   |  |
| Random-2    | 275         | 60                  | 200                   |  |
| Random-3    | 275         | 60                  | 200                   |  |
| Random-4    | 300         | 20 → 60             | 200                   |  |

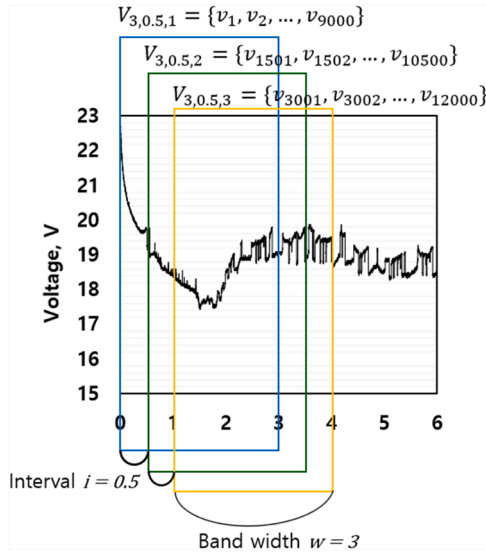
some of them were not feasible; thus, 15 feasible conditions were selected. Four additional feasible experiments were also added (denoted as “Random” in Table 1). As a result, 19 bead-on-plate tests were conducted. During the experiments, the other WAAM process variables were fixed, such as the shield gas (100%) and arc length (5 mm). Table 1 lists the experimental conditions. Data from the fifteen and four experiments were used for training and testing the proposed algorithm, respectively.

Three types of data were obtained for each experiment: voltage, video, and bead image data. The voltage value was measured and

collected by a sensor every 1/3000 s, a nearly real-time collection, and the video data monitored the process at 50 frames per second (fps). The last column of Table 1 shows the images for the Mo beads obtained from the 19 experiments.

### 3.2. Modules of the proposed algorithm

This section describes the image conversion module, CNN prediction module, and real-time monitoring module, which are the core of the

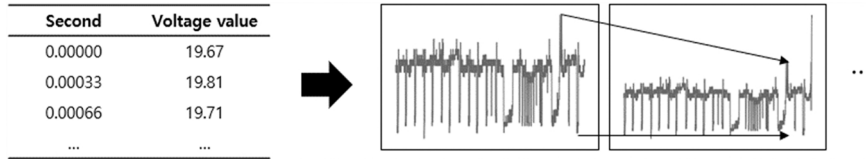


**Fig. 3.** The horizontal and vertical axes denote  $T$  and  $v_m$ , respectively, at  $T$ . Because  $w = 3$ , each  $V_{w,i,k}$  element includes 9000 voltage values.  $V_{3,0.5,1}$  is the first voltage value dataset; thus, it includes  $v_1, v_2, \dots, v_{9000}$ . Because  $i = 0.5$ ,  $V_{3,0.5,2}$  includes  $v_{1501}, v_{1502}, \dots, v_{10500}$ . Note that any two consecutive voltage value datasets share 7500 voltage values because  $w = 3$  and  $i = 0.5$ .

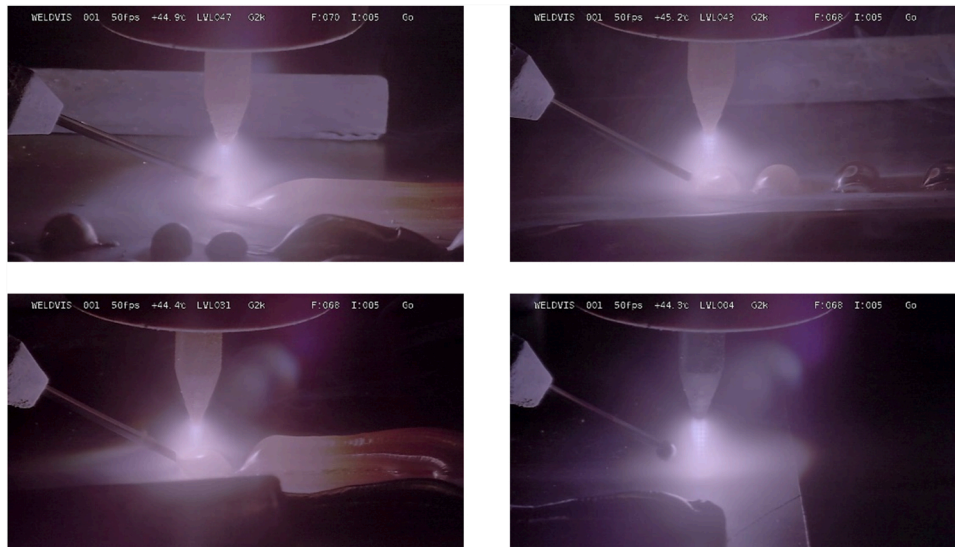
proposed algorithm.

### 3.2.1. Image conversion module

The image conversion module converts the voltage values into



**Fig. 4.** Compared with the left voltage image, the right voltage image has a larger maximum value of  $v_m$ . Thus, the range of the vertical axis of the right voltage image becomes wider to include a larger voltage value while maintaining the same image size. This adjustment is required to have all the  $v_m$  data while maintaining the same image size.



**Fig. 5.** The upper left figure is an example of a normal frame where the shape of the Mo bead is stable. The upper right shows an abnormal type where the Mo bead is cut off from the earlier beads. In the lower left figure, the right part of the arc appears to be normal, while the left part is abnormal. The lower right figure is a frame in which the WAAM process begins. The two figures in lower side are neither normal nor abnormal; thus, they are classified as neutral.

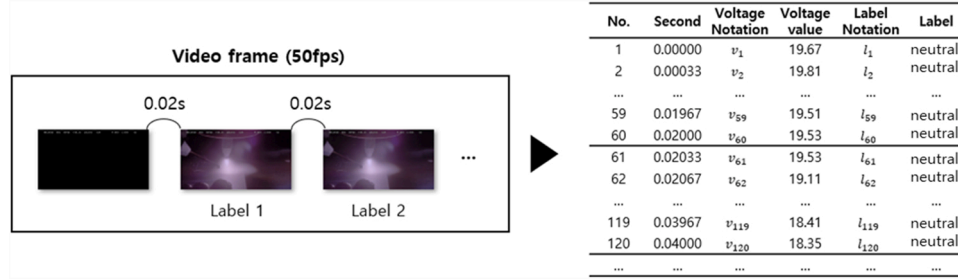


Fig. 6. All the 120  $v_m$ s are labeled as neutral (i.e.,  $l_1, \dots, l_{120} = \text{neutral}$ ) because the label of the two frames are neutral.

Subsequently, the size of the images can differ because it is determined by the minimum and maximum values of  $v_m$ . However, CNN requires all images to have the same size. Therefore, the vertical axes of the images are adjusted such that every image has the same size of  $224 \times 224$  pixels (Fig. 4).

The voltage value data were divided into two groups and separately converted into voltage images since a training dataset was required to develop the CNN classifier and a test dataset to validate the CNN classifier. The first group, used for training, included the voltage value data collected from 15 experiments, denoted as DOE-1, 2, ..., 15 in Table 1. The second, used for testing, had the voltage value data collected from four experiments, denoted as Random-1, 2, 3, 4 in Table 1. The bandwidth was set to 4 s (i.e.,  $w = 4$ ) for both groups, so each voltage image had  $3000 \times 4 = 12,000$   $v_m$ s. The parameter  $i$  was set to 0.2 and 0.02 for the training and test datasets, respectively. Therefore, the voltage image was generated every 0.2 s in the training dataset and every 0.02 s in the test dataset. As a result, 537 and 2505 voltage images were obtained for the training and test datasets, respectively.

It is worth noting that the number of the training data is larger than the testing data. In the proposed method, we collected training data from 15 beads (DOE-1, 2, ..., 15 in Table 1) and testing data from 4 beads (Random-1, 2, 3, 4 in Table 1). Thus, it is natural that the number of training data is larger than the testing. However, the number of data is also mainly determined by interval  $i$  which indicates the time gap between two consecutive images of voltage values. For a given time, as  $i$  decreases, the number of images increases. For example, for 5 s, only 5 images are generated by setting  $i = 1$  s while 50 images are generated by setting  $i = 0.1$  s. In this case, we set small  $i$  value for generating many testing data because this makes the testing environment close to real-time monitoring.

### 3.2.2. CNN prediction module

As previously mentioned, the converted voltage image is the input to the CNN prediction module. Each converted voltage image is classified to determine whether the WAAM process is normal or abnormal. For this purpose, a CNN classifier should be developed in advance. The CNN classifier is trained and tested by the training and test dataset which

were generated by labeling the converted voltage images based on the video data. Labeling consists of three sequential labeling tasks for video frame,  $v_m$ , and voltage image data.

First, the video frames are labeled. The video data has several frames, and each frame contains information on whether the process is normal or abnormal. Video labeling is conducted manually by classifying all frames of the video data. Each frame is categorized into three types: *normal*, *abnormal*, and *neutral* as shown in Fig. 5.

When the video frames are labeled,  $v_m$  is recognized by the labels of the video data. The label of  $v_m$  is denoted as  $l_m$ . Similar to video labeling,  $v_m$  is classified into the same three types. Note that  $v_m$  is collected every 1/3000 s, while the video frame is collected every 1/50 s. Thus, each video frame includes 60  $v_m$ s. These 60  $v_m$ s are labeled as the same type of the associated frame. Fig. 6 illustrates labeling 120  $v_m$ s included in the first two frames.

Next, voltage images are labeled. In the image conversion module, each  $V_{w,i,k}$  is converted into a voltage image. There is a one-to-one conversion between  $V_{w,i,k}$  and the voltage image; thus, if  $V_{w,i,k}$  is labeled, its voltage image is labeled the same. Let us denote  $L_{w,i,k}$  as a set of  $l_m$ s in  $V_{w,i,k}$ .  $V_{w,i,k}$  is labeled based on  $L_{w,i,k}$ . As described in Section 3.2.1, two sets of voltage images were generated: the training and test. If  $V_{w,i,k}$  is for training (test), it is labeled according to Algorithm 1 (2). In Algorithm 1,  $V_{w,i,k}$  is labeled as normal (abnormal) when all of  $l_m$ s in  $V_{w,i,k}$  are normal (abnormal). In other words, if any  $l_m$  is neutral or two types of labels are mixed,  $V_{w,i,k}$  is excluded from the training dataset. The difference between Algorithms 1 and 2 is that Algorithm 1 assigns labels to a voltage image only when all  $l_m$  in  $V_{w,i,k}$  are either normal or abnormal. For example, when  $i = 4$ ,  $V_{w,i,k}$  includes 12,000  $l_m$ s from 200 video frames and all the 12,000  $l_m$ s are either normal or abnormal. Generally, CNN models are known to exhibit good classification performance when the training data have clear patterns of objects to be classified. These pure images better depict the behavior of the time-series voltage data generated when the WAAM process is either normal or abnormal. Ultimately, this improves the classification performance of the CNN model.

**Algorithm 1.** Labeling for the training dataset.

---

```

1: procedure TrainDataSetLabeling( $L_{w,i,k}$ )
2:   if  $L_{w,i,k}$  only contains 'Normal' THEN
3:     Label = 'Normal'
4:   else if  $L_{w,i,k}$  only contains 'Abnormal' THEN
5:     Label = 'Abnormal'
6:   else
7:     Remove  $V_{w,i,k}$ 
8:   endif
9: return Label

```

---



**Algorithm 2.** Labeling for the test dataset.

---

```

1: procedure TestDataSetLabeling( $L_{w,i,k}$ )
2:   if  $L_{w,i,k}$  contains ‘Normal’ more than ‘Abnormal’
THEN
3:     Label = ‘Normal’
4:   else  $L_{w,i,k}$  contains ‘Abnormal’ more than
‘Normal’ THEN
5:     Label = ‘Abnormal’
6:   endif
7: return Label

```

---

As described in Section 3.2.1, 537 and 2505 voltage images were obtained for the training and test datasets, respectively. We labeled these images using three labeling tasks. As a result, the 537 voltage images were labeled as 390 normal and 147 abnormal images. Similarly, the 2505 voltage images were labeled as 1495 normal and 1010 abnormal images. We developed and validated a CNN classifier based on these training and test datasets. DenseNet169 was employed as the CNN model structure, Adam as an optimizer, with 16 batches, 30 epochs, and categorical cross-entropy as a loss function to develop the CNN classifier.

We used 70% of the training dataset to train the CNN classifier and the remaining 30% of the training dataset as the validation dataset. CNN model weights were selected at the 29th epoch, where the accuracy of the validation dataset reached 95%. Subsequently, the classification performance of the trained CNN classifier was evaluated using the test dataset. The CNN classifier achieved 96% accuracy for the test dataset. In addition, it was assessed using the four indices of accuracy, precision, recall, and f1-score. Accuracy is the ratio between the number of classified images and the number of images. The total number of images was 2505, and 2405 were correctly classified; thus, the accuracy was calculated as 0.96 ( $= 2405 / 2505$ ). The other three indices were calculated for each label. Precision is regarded as the ratio between the number of correctly classified images and the number of classified images for each label. The recall is the ratio between the number of correctly classified images and the number of images for each label. The f1-score is a harmonized mean of the precision and recall for each label. Support denotes the number of images for each label. Table 2 lists the three indices of the CNN classifier for the normal and abnormal labels.

**Table 2**  
Performance of the CNN classifier.

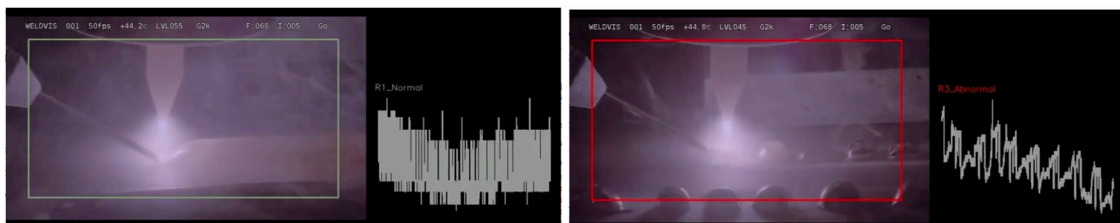
| Label    | Precision | Recall | f1-score | Support |
|----------|-----------|--------|----------|---------|
| Normal   | 0.94      | 0.99   | 0.97     | 1495    |
| Abnormal | 0.99      | 0.90   | 0.95     | 1010    |

For a normal label, 1405 images among 1495 normal images were correctly classified; thus, the precision was 0.94 ( $= 1405 / 1495$ ). Additionally, 1420 images were classified as normal among 2505 images; thus, the recall was 0.99 ( $= 1405 / 1420$ ). The f1-score was calculated as 0.97 using the harmonized mean of 0.94 and 0.99. The three indices of the abnormal scenario were calculated in a similar manner. All the indices were larger than 0.90, which meant that the CNN classifier had a good classification performance. Among the indices, recall of the abnormal label had the lowest value of 0.90. This will be investigated in Section 3.3.

### 3.2.3. Real-time monitoring module

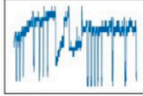
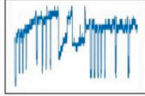
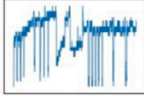
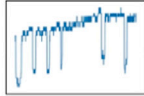
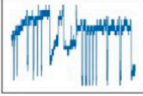
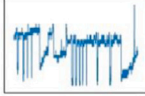
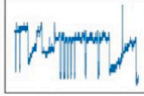
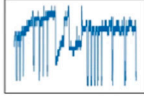
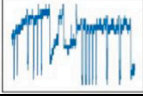
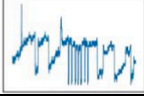
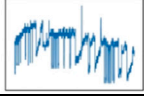
Real-time monitoring is performed simultaneously with image conversion and CNN prediction modules. Whenever the voltage image is generated from the image conversion module, the CNN prediction module classifies it as normal or abnormal through the CNN classifier, trained and validated in advance. The real-time monitoring module provides the results of the CNN prediction model in real-time through a video on a dashboard. Fig. 7 show snapshots of the video on the dashboard. The video file is attached in the end of this paper. It shows real-time monitoring performed by classifying the 2505 voltage images of the test dataset. Whenever the bead shape became abnormal in the video, the square became red. Additionally, the voltage image became unstable when the process was abnormal. Overall, we conclude that the proposed algorithm performed well.

It should be noted that real-time monitoring starts after the first 4 s in the video. This is because the bandwidth was set to 4 s. After the first 4 s, the first voltage image was generated, and real-time monitoring started. Because  $i = 0.02$ , the voltage image was generated every 0.02 s; thus, the real-time monitoring was updated every 0.02 s. Notably, there was a time gap between the time when the WAAM process changed from normal to abnormal and when the square changed from green to red. Suppose that  $w = 4$  and the WAAM process changes from normal to abnormal at time point of 10 s. In such a case, the voltage image at the time point of 10 s includes  $v_m$ s have been collected from time point of 6 s



**Fig. 7.** The left parts of both figures show the video frames, and the right part show the voltage image data. The square in the video frame indicates whether the WAAM is normal or abnormal, with green (left figure) and red (right figure) colors, respectively.

**Table 3**  
Image conversion by dataset condition.

| Interval | Images                                                                            |                                                                                   |                                                                                   | Bandwidth | Images                                                                             |
|----------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-----------|------------------------------------------------------------------------------------|
| 0.02 s   |  |  |  | 0.5 s     |  |
| 0.2 s    |  |  |  | 2 s       |  |
| 1 s      |  |  |  | 4 s       |  |

to time point 10 s. During the 4 s, the WAAM process is normal; thus, the voltage image at the time point of 10 s shows a normal time-series pattern of  $v_m$ . Therefore, CNN classifier classifies the image as normal. As time passes, newly generated voltage images have more  $v_m$ s from the abnormal process; therefore, their time-series pattern of  $v_m$  become unstable. The CNN classifier classifies the voltage image as abnormal when the unstable part becomes larger than the stable part. This is why a time gap exists between the two images and explains why the recall score of the abnormal label in Table 3 has a low value.

### 3.3. Sensitivity analysis for $i$ and $w$

As described in Section 3.2, we set  $i = 0.02$ ,  $w = 4$  for the training dataset, and  $i = 0.2$ ,  $w = 4$  for the test dataset. The two parameters  $i$  and  $w$  are important because they affect the classification performance and

computational costs. Thus, it is investigated by testing the proposed method with respect to various combinations of  $i$  and  $w$ . As previously mentioned, the training and test datasets were prepared through three labeling tasks: video labeling,  $v_m$  labeling, and voltage image labeling. The two parameters are related to the voltage image labeling task because they determine the interval and bandwidth.

As shown in Fig. 3, a voltage image is generated by sliding the time window. When  $i$  is small, the interval between the time windows is small; thus, the generated images overlap significantly. The images then become similar to each other because they share a large number of  $v_m$  values. This makes it difficult to extract features when training a CNN classifier. Additionally, a small  $i$  generates a large number of images, which increases the learning time but helps train the CNN classifier. Thus, a small  $i$  has both advantages and disadvantages. The other parameter  $w$  determines the length of the time window. When  $w$  is large,

**Table 4**  
Dataset.

| Dataset | Training dataset |     |                                |          |       | Test dataset |     |                                |          |       |
|---------|------------------|-----|--------------------------------|----------|-------|--------------|-----|--------------------------------|----------|-------|
|         | Condition        |     | The number of generated images |          |       | Condition    |     | The number of generated images |          |       |
|         | $i$              | $w$ | Normal                         | Abnormal | Total | $i$          | $w$ | Normal                         | Abnormal | Total |
| DS1     | 0.02             | 0.5 | 5506                           | 2357     | 7863  | 0.02         | 0.5 | 2072                           | 1133     | 3205  |
| DS2     | 0.1              | 0.5 | 1106                           | 466      | 1572  | 0.02         | 0.5 | 2072                           | 1133     | 3205  |
| DS3     | 0.2              | 0.5 | 543                            | 245      | 788   | 0.02         | 0.5 | 2072                           | 1133     | 3205  |
| DS4     | 0.5              | 0.5 | 215                            | 98       | 313   | 0.02         | 0.5 | 2072                           | 1133     | 3205  |
| DS5     | 1                | 0.5 | 110                            | 47       | 157   | 0.02         | 0.5 | 2072                           | 1133     | 3205  |
| DS6     | 0.02             | 1   | 5205                           | 2166     | 7371  | 0.02         | 1   | 1910                           | 1195     | 3105  |
| DS7     | 0.1              | 1   | 1049                           | 424      | 1473  | 0.02         | 1   | 1910                           | 1195     | 3105  |
| DS8     | 0.2              | 1   | 508                            | 230      | 738   | 0.02         | 1   | 1910                           | 1195     | 3105  |
| DS9     | 0.5              | 1   | 208                            | 86       | 294   | 0.02         | 1   | 1910                           | 1195     | 3105  |
| DS10    | 1                | 1   | 106                            | 43       | 149   | 0.02         | 1   | 1910                           | 1195     | 3105  |
| DS11    | 0.02             | 1.5 | 4950                           | 1995     | 6945  | 0.02         | 1.5 | 1764                           | 1241     | 3005  |
| DS12    | 0.1              | 1.5 | 989                            | 399      | 1388  | 0.02         | 1.5 | 1764                           | 1241     | 3005  |
| DS13    | 0.2              | 1.5 | 492                            | 203      | 695   | 0.02         | 1.5 | 1764                           | 1241     | 3005  |
| DS14    | 0.5              | 1.5 | 197                            | 80       | 277   | 0.02         | 1.5 | 1764                           | 1241     | 3005  |
| DS15    | 1                | 1.5 | 98                             | 40       | 138   | 0.02         | 1.5 | 1764                           | 1241     | 3005  |
| DS16    | 0.02             | 2   | 4709                           | 1850     | 6559  | 0.02         | 2   | 1657                           | 1248     | 2905  |
| DS17    | 0.1              | 2   | 942                            | 369      | 1311  | 0.02         | 2   | 1657                           | 1248     | 2905  |
| DS18    | 0.2              | 2   | 460                            | 196      | 656   | 0.02         | 2   | 1657                           | 1248     | 2905  |
| DS19    | 0.5              | 2   | 187                            | 74       | 261   | 0.02         | 2   | 1657                           | 1248     | 2905  |
| DS20    | 1                | 2   | 91                             | 42       | 133   | 0.02         | 2   | 1657                           | 1248     | 2905  |
| DS21    | 0.02             | 3   | 4284                           | 1620     | 5904  | 0.02         | 3   | 1570                           | 1135     | 2705  |
| DS22    | 0.1              | 3   | 865                            | 316      | 1181  | 0.02         | 3   | 1570                           | 1135     | 2705  |
| DS23    | 0.2              | 3   | 433                            | 159      | 592   | 0.02         | 3   | 1570                           | 1135     | 2705  |
| DS24    | 0.5              | 3   | 168                            | 68       | 236   | 0.02         | 3   | 1570                           | 1135     | 2705  |
| DS25    | 1                | 3   | 86                             | 34       | 120   | 0.02         | 3   | 1570                           | 1135     | 2705  |
| DS26    | 0.02             | 4   | 3930                           | 1427     | 5357  | 0.02         | 4   | 1495                           | 1010     | 2505  |
| DS27    | 0.1              | 4   | 793                            | 280      | 1073  | 0.02         | 4   | 1495                           | 1010     | 2505  |
| DS28    | 0.2              | 4   | 390                            | 147      | 537   | 0.02         | 4   | 1495                           | 1010     | 2505  |
| DS29    | 0.5              | 4   | 155                            | 59       | 214   | 0.02         | 4   | 1495                           | 1010     | 2505  |
| DS30    | 1                | 4   | 81                             | 28       | 109   | 0.02         | 4   | 1495                           | 1010     | 2505  |



**Table 5**  
Results of the model selection experiments.

|         | Indicators      | ResNet50     | ResNet101    | DenseNet121  | DenseNet169  |
|---------|-----------------|--------------|--------------|--------------|--------------|
| DS6     | Accuracy        | 0.847        | 0.896        | 0.886        | 0.900        |
|         | Learning time   | 1729.28      | 2900.20      | 1849.09      | 2254.93      |
|         | Prediction time | 5.42 (0.002) | 9.39 (0.004) | 7.03 (0.003) | 8.44 (0.003) |
| DS16    | Accuracy        | 0.803        | 0.871        | 0.844        | 0.898        |
|         | Learning time   | 1523.56      | 2553.74      | 1646.17      | 1992.76      |
|         | Prediction time | 5.67 (0.002) | 8.86 (0.004) | 5.71 (0.002) | 7.55 (0.003) |
| DS21    | Accuracy        | 0.835        | 0.854        | 0.889        | 0.920        |
|         | Learning time   | 1366.24      | 2288.88      | 1476.30      | 1821.41      |
|         | Prediction time | 5.31 (0.002) | 8.60 (0.004) | 6.04 (0.003) | 6.97 (0.003) |
| Average | Accuracy        | 0.828        | 0.874        | 0.873        | 0.906        |
|         | Learning time   | 1539.69      | 2580.94      | 1657.19      | 2023.03      |
|         | Prediction time | 5.47         | 8.95         | 6.26         | 7.65         |

**Table 6**  
Results of the dataset selection.

| Data set | Interval | Bandwidth | Accuracy | Learning Time | Prediction Time |
|----------|----------|-----------|----------|---------------|-----------------|
| DS1      | 0.02     | 0.5       | 0.887    | 3855.04       | 7.19 (0.002)    |
| DS2      | 0.1      | 0.5       | 0.889    | 936.01        | 7.11 (0.002)    |
| DS3      | 0.2      | 0.5       | 0.899    | 569.95        | 7.20 (0.002)    |
| DS4      | 0.5      | 0.5       | 0.677    | 328.25        | 7.19 (0.002)    |
| DS5      | 1        | 0.5       | 0.647    | 249.14        | 7.17 (0.002)    |
| DS6      | 0.02     | 1         | 0.917    | 3651.93       | 6.95 (0.002)    |
| DS7      | 0.1      | 1         | 0.934    | 893.33        | 6.96 (0.002)    |
| DS8      | 0.2      | 1         | 0.857    | 540.86        | 6.97 (0.002)    |
| DS9      | 0.5      | 1         | 0.841    | 322.65        | 6.93 (0.002)    |
| DS10     | 1        | 1         | 0.488    | 246.33        | 6.89 (0.002)    |
| DS11     | 0.02     | 1.5       | 0.887    | 3583.95       | 6.71 (0.002)    |
| DS12     | 0.1      | 1.5       | 0.881    | 835.37        | 6.74 (0.002)    |
| DS13     | 0.2      | 1.5       | 0.680    | 491.06        | 6.71 (0.002)    |
| DS14     | 0.5      | 1.5       | 0.715    | 298.28        | 6.75 (0.002)    |
| DS15     | 1        | 1.5       | 0.587    | 235.01        | 6.69 (0.002)    |
| DS16     | 0.02     | 2         | 0.905    | 3218.37       | 6.47 (0.002)    |
| DS17     | 0.1      | 2         | 0.849    | 823.78        | 6.52 (0.002)    |
| DS18     | 0.2      | 2         | 0.839    | 496.44        | 6.55 (0.002)    |
| DS19     | 0.5      | 2         | 0.571    | 300.19        | 6.53 (0.002)    |
| DS20     | 1        | 2         | 0.571    | 235.16        | 6.52 (0.002)    |
| DS21     | 0.02     | 3         | 0.871    | 3023.32       | 6.07 (0.002)    |
| DS22     | 0.1      | 3         | 0.880    | 754.65        | 6.06 (0.002)    |
| DS23     | 0.2      | 3         | 0.885    | 464.73        | 6.06 (0.002)    |
| DS24     | 0.5      | 3         | 0.581    | 291.14        | 6.10 (0.002)    |
| DS25     | 1        | 3         | 0.581    | 233.64        | 6.11 (0.002)    |
| DS26     | 0.02     | 4         | 0.937    | 2679.58       | 5.64 (0.002)    |
| DS27     | 0.1      | 4         | 0.940    | 706.76        | 5.70 (0.002)    |
| DS28     | 0.2      | 4         | 0.958    | 440.64        | 5.65 (0.002)    |
| DS29     | 0.5      | 4         | 0.598    | 280.25        | 5.61 (0.002)    |
| DS30     | 1        | 4         | 0.538    | 217.13        | 5.61 (0.002)    |

each generated image includes many  $v_m$  values; so it is useful for training the classifier. However, a large  $w$  would cause the images to overlap, and the generated images become similar. This image overlapping problem is caused not only by a small  $i$  but also by a large  $w$ . So the optimal values for  $i$  and  $w$  should be investigated.

In addition,  $i$  and  $w$  also affect the real-time monitoring module. As previously mentioned, real-time monitoring begins after the first  $w$  seconds. Thus, as  $w$  increases, real-time monitoring starts with a delay. The other parameter  $i$  determines the time interval between the voltage images. The WAAM process is determined to be normal or abnormal in the proposed algorithm whenever the voltage image is generated. When  $i$  used in real-time monitoring is small, the voltage images are frequently generated; thus, the monitoring becomes closer to real-time. However, a small  $i$  value would require a tremendous computational effort because the CNN prediction and real-time monitoring modules are performed and updated whenever the voltage image is generated.

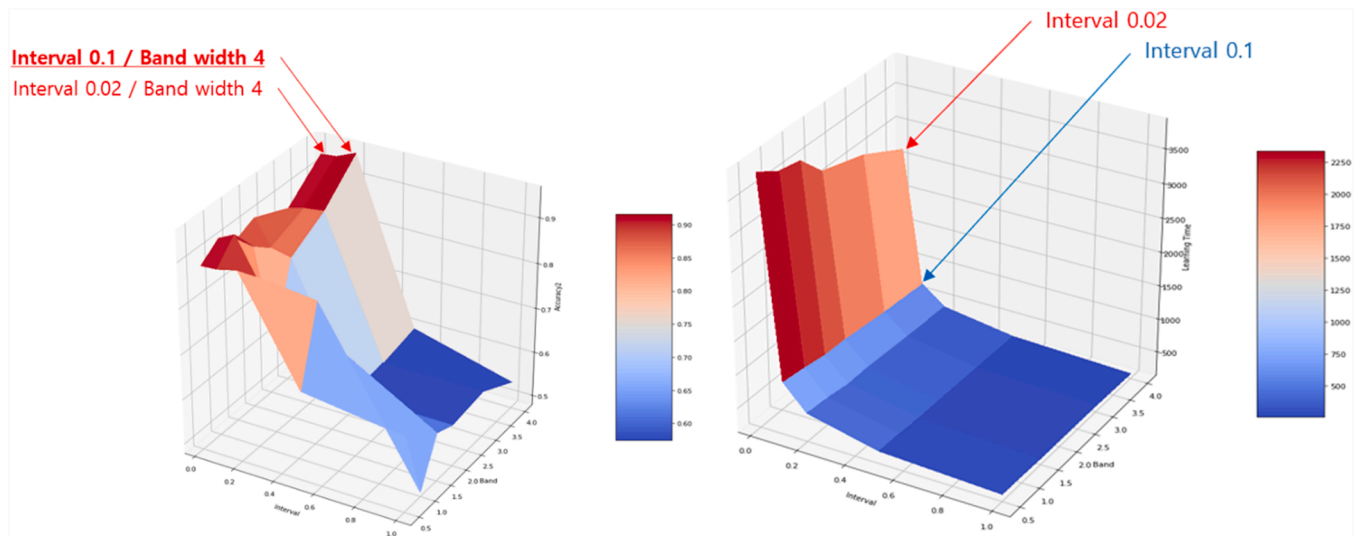
Table 3 shows examples of the voltage images according to several interval and bandwidth values. The nine left images were generated by setting  $i = 0.02$  and  $0.2$  while maintaining  $w = 2$ . When  $i = 0.02$ , the

three images significantly overlapped. In contrast, they overlapped less when  $i = 1$ . The right three images were generated by setting  $w = 0.5$ ,  $2$ , and  $4$  while maintaining  $i = 2$ . When  $w = 0.5$ , only a small part of the time-series plot was shown. In contrast, when  $w = 4$ , a large part of the time-series plot was shown. Note that the images had the same size at  $224 \times 224$  pixels, although  $w$  differed.

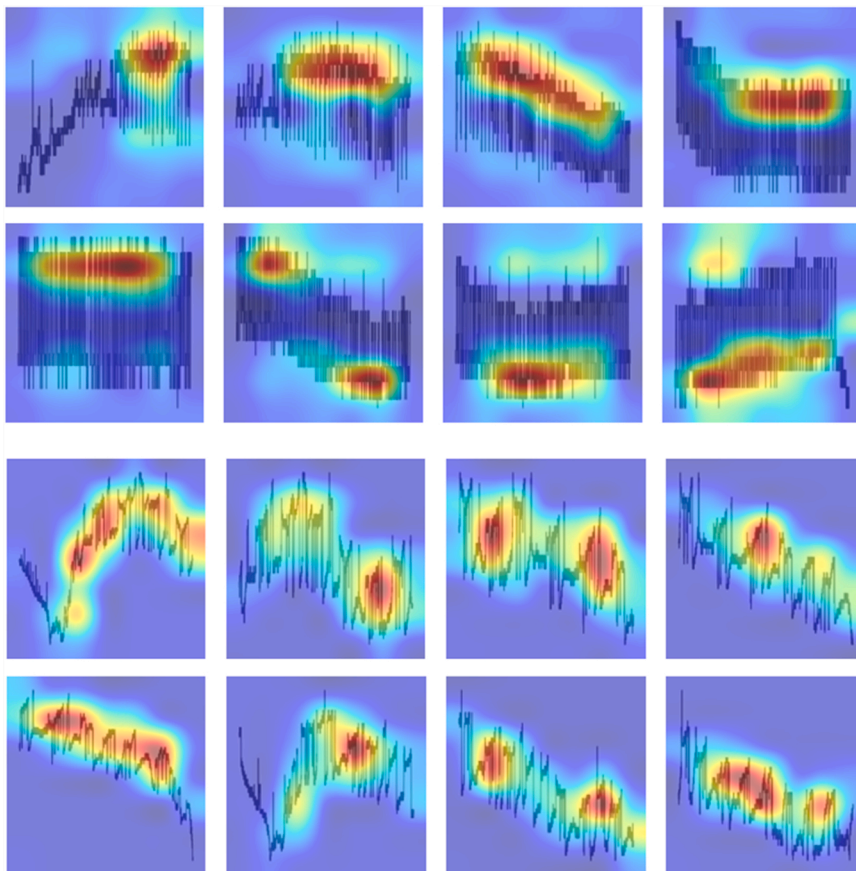
To investigate the effect of  $w$  and  $i$ , five values;  $0.02$ ,  $0.1$ ,  $0.2$ ,  $0.5$ , and  $1$  for  $i$  and six values;  $0.5$ ,  $1$ ,  $1.5$ ,  $2$ ,  $3$ , and  $4$  for  $w$  were set. Therefore, a total of 30 combinations of  $i$  and  $w$  values were obtained. First, voltage images were generated for the training dataset according to the 30 combinations of  $i$  and  $w$  (left part of Table 4). The voltage values collected from DOE 1, 2, ..., 15 in Table 1, were converted into training voltage images. We denote the 30 sets of the generated voltage images as DS1, DS2, ..., DS30. As expected, the number of generated images decreased as  $i$  increased. The number of images decreased slightly as  $w$  increased. The first image was generated after the first  $w$  seconds because it takes  $w$  seconds to generate. Thus, as  $w$  decreased, the first image appeared earlier, which meant that a larger number of images were generated. However, this increase was not significant compared with the increase due to  $i$  (Table 4). For the test datasets, we fixed  $i = 0.02$  and changed  $w$ , as shown in Table 4.

Second, we generated voltage images for the test dataset according to six combinations of  $i$  and  $w$  (right part of Table 4). In Table 4,  $i$  is fixed at a small value of  $0.02$ , and only  $w$  has six values. This was done to generate many images for the test dataset, which is useful for real-time monitoring. By setting  $i$  as a small value of  $0.02$  s, the images are generated every  $0.02$  s; Thus, the proposed algorithm detects the abnormal process every  $0.02$  s, which is very close to real-time. For example, the training datasets of DS28 and DS26 were generated by setting  $i = 0.2$  s and  $0.02$  s, respectively, for a same  $w$  value; thus, DS28 is a subset of DS26. In case of DS26, the number of training data is larger than that of testing data because both training and testing datasets were generated from the same interval (i.e.,  $i = 0.02$  s). In DS28, we increased  $i$  from  $0.02$  to  $0.2$  when generating the training data but did not increase it when generating testing data in order to make the testing environment close to real-time monitoring. After preparing all the datasets, we trained and tested the CNN classifiers on each dataset.

Before training the CNN classifier, an appropriate CNN structure should be selected in advance. We considered four CNN structures, ResNet50, ResNet101, DenseNet121, and DenseNet50 as alternatives because they are available in Keras applications. It was desirable to apply the four CNN structures to all 30 datasets. However, this involved high computational costs, resulting in long training and testing time. Applying the four structures for the 30 datasets meant 120 CNN classifiers were to be trained. Therefore, we selected only DS6, DS16, and DS21 among the 30 datasets and applied the four CNN structures to the selected datasets. Specifically, 70% of the training data for each dataset were used to train the CNN classifier and the other 30% as a validation set. During the training, the epoch increased, and the accuracy of the validation set changed accordingly. The optimal CNN weights were



**Fig. 8.** Accuracy (left) is primarily affected by  $i$  rather than  $w$ , and it drastically decreases as  $i$  increases from 0.2 to 0.5. This means that when  $i$  is larger than 0.2, the number of images is insufficient to train the CNN classifiers with high accuracy. Therefore, it is desirable to set  $i$  to less than 0.2. However, the learning time (left) drastically increased when  $i$  is less than 0.1. Consequently, it is recommended to set  $i$  between 0.1 and 0.2 when considering the tradeoffs between accuracy and learning time. In contrast, as  $w$  increased, the accuracy increased. However, a large  $w$  had the disadvantage of a late start of real-time monitoring. Thus, when determining  $w$ , both the accuracy and starting time of real-time monitoring should be considered.



**Fig. 9.** The eight images in upper side and the other eight images in lower side are classified as normal and abnormal, respectively. In the normal cases, the red regions were located either in the upper or lower parts of the time-series waveforms in the images, and they appeared stable without exhibiting drops. In contrast, the red regions in the abnormal cases were located in the middle of the time-series waveforms in the images, and they exhibited large drops compared with the normal figures.

determined when the accuracy of the validation set was the highest. Subsequently, the test dataset was used to evaluate the classification performance of the classifier.

Table 5 shows the accuracy, learning, and prediction times of the CNN classifiers obtained by applying the four CNN structures. The learning time is the elapsed time for training the CNN classifier, and the

prediction time is the time for classifying the voltage images of the test dataset. Values in parentheses in prediction time in Table 5 indicate the average time to predict a single voltage image. All four CNN structures exhibited a robust performance with respect to the three datasets. ResNet50 was first excluded because it exhibited the lowest accuracy. Additionally, ResNet101 was also omitted because it showed the most

prolonged learning and prediction time. DenseNet169 has better accuracy than DenseNet121, whereas DenseNet121 is faster in learning and prediction. Finally, DenseNet169 was selected because it significantly increases the accuracy while only slightly sacrificing the learning and prediction times compared with DenseNet121.

The next step was training the CNN classifiers for the 27 datasets while excluding DS6, DS16, and DS21, because they were already fitted to select the appropriate CNN structure. It was observed that DenseNet169 was suitable for DS6, DS16, and DS21; thus, it was used again to train the CNN classifiers for the other 27 datasets. Table 6 lists the accuracy, learning, and prediction times of the CNN classifiers for the 30 datasets. They were fitted by applying DenseNet169. In terms of interval  $i$ , the number of voltage images and learning time increased as  $i$  decreased, as expected. This is because a more significant number of voltage images were generated as  $i$  decreased. In terms of bandwidth  $w$ , the accuracy increased as  $w$  increased because a voltage image with a large  $w$  has many  $v_m$  values. As the voltage image includes more  $v_m$  values, it includes a larger part of the time-series plot. This is useful for improving classification performance. Moreover,  $w$  slightly affects the learning time. As  $w$  increases, the first image is generated later; thus, fewer images are generated. A large  $w$  leads to short learning time; yet delays the start of real-time monitoring. Additionally, the prediction time decreases as  $w$  increases for the same reason. However, the average elapsed time of the prediction time does not change according to  $i$  and  $w$ .

It was observed that the overall behavior of accuracy, learning, and prediction times according to  $i$  and  $w$  was reasonable. Next, we attempted to determine the optimal  $i$  and  $w$  for which the accuracy, learning, and prediction time are desirable. The prediction time was not considered because the average time is not affected by  $i$  and  $w$ . Fig. 8 shows the accuracy (left) and learning time (right) according to  $i$  and  $w$ . In this example, we recommend setting  $i$  between 0.1 and 0.2 and  $w$  to 4 where both accuracy and learning time were satisfactory.

On the other hand, it is interesting to compare the proposed method with the existing CNN method that does not include 1D data. Recently, [26] suggested another real-time monitoring algorithm using CNN. In their work, the CNN model was trained from the 2D data which are same data that we used. [26] generated 2D images with interval  $i = 0.033$  s/frame and test their CNN model. As a result, their CNN model showed 96% of accuracy. In case of the proposed method, we generated 30 datasets (denoted as DS1,2,...,30 in Table 4) by setting various  $i$  and  $w$  values. Among them, DS21, which was generated with  $i = 0.02$  s/frame, is most similar to the data [26] used. As a result, the proposed CNN model showed 93.7% of accuracy as reported in Table 4. Both CNN models employed same model structure of DenseNet169 and similar intervals and were applied; thus, we believe that the comparing two models is fair. Our accuracy is slightly less than [26] and this is somewhat expected result because [26]'s model works based on 2D video frame data while the proposed method works based on simple 1D voltage data. Although the classification accuracy of the proposed method is slightly smaller, the proposed method has advantages of using simple 1D data.

### 3.4. Validation using Grad-CAM

As described in Sections 3.2 and 3.3, the CNN classifier with a good classification performance was obtained. We investigated the reason behind the good classification performance of CNN classifiers and validated it using a class activation mapping (CAM) method. When the CNN classifier determines each voltage image, it focuses on particular parts that express the features of normal and abnormal classes. We employed a particular CAM method called grad-CAM, which highlights these parts of the images, which are considered more important than the other regions.

Fig. 9 shows examples of grad-CAM for the CNN classifier that was trained in Section 3.2. Specifically, grad-CAM was applied to the last convolutional layer. It was implemented using `visualize_cam`, which is a Keras visualization function. `Visualize_cam` expresses the result of grad-CAM in color in the voltage image. Red regions are important regions where the CNN classifier focuses on classifying the voltage image. In contrast, the blue regions represent the regions in which the CNN classifier does not concentrate. Overall, we concluded that the location and shape of the red regions of the normal and abnormal figures were markedly different, which implied that the CNN classifiers were adequately trained.

## 4. Concluding remarks

In this study, a CNN-based real-time monitoring algorithm was developed using the voltage data generated during fabricating Mo beads by the WAAM process. The algorithm consists of three modules: image conversion, CNN prediction, and real-time monitoring. The image conversion module converts the voltage values into voltage images in a time-series waveform. The CNN prediction module categorizes each voltage image into normal or abnormal to detect whether the WAAM process is normal or abnormal. Finally, the real-time monitoring module expresses the results of the CNN prediction model on a real-time dashboard. The performance of the proposed algorithm is highly dependent on two parameters,  $i$  and  $w$ . Therefore, a sensitivity analysis was conducted for  $i$  and  $w$  to determine the effects of the two parameters, based on which guidelines for setting them were obtained. Finally, the CNN classifiers were validated by applying grad-CAM to visualize regions that are important for classification. It was concluded that the CNN classifiers were adequately trained because they captured the critical regions in the voltage images.

It is interesting to discuss applicability of classical machine learning (ML) methods to develop the classifier. In the proposed method, CNN was employed as the classifier because it does not require feature extraction while the classical ML methods do. In the WAAM process, voltage data are collected on a real-time basis and the process is judged as normal or abnormal based on the voltage data. Thus, the algorithm should be able to classify the time-series data into normal or abnormal. When learning the classical ML models from the time-series data, features should be determined by the users, and then, the models classify the time-series data based on the features. Thus, the performance of the classical ML models largely depends on the feature extraction. In contrast, CNN does not require the feature extraction. The proposed method converts the time-series data into 2D images and CNN model classifies the 2D images. Because CNN automatically extracts spatial features of 2D images, users are not required to determine the features [27]. In our WAAM process, we had difficulties in defining features of the voltage data, thus, we used the CNN model for time-series classification. On the other hand, this feature-extraction-free approach has a disadvantage over the classical ML methods in that it is a "black-box" model. The CNN model cannot explain why it classifies each time-series data into normal or abnormal. In contrast, the classical ML models, for example, support vector machine (SVM), can explain the reason because SVM estimates coefficient for each feature. Thus, we can identify which features are critical in classifying the WAAM process based on the estimated coefficients. In order to overcome this disadvantage, we additionally conducted grad-CAM in Section 3.4, to highlight regions of the images, which serve as critical features in the classification.

From the data mining perspective, our problem can be considered as a time-series classification problem. Instead of CNN, meta learning can be employed to train the 1D data (time-series data) classification model. Main advantage of the meta learning is good performance with a small learning set. In spite of this advantage, meta learning methods have not



been applied to time-series data classification in the WAAM process so far. We believe that using meta learning methods can be very good when the learning data is small. However, even if the meta learning methods are applied, we believe that our approach of mapping 1D to 2D and training 2D classification model is still useful because meta learning methods show good performance in image classification problems.

In future research, it will be necessary to generate new types of waveforms that are not used for learning. There is a limit to adding a new type of waveform each time, and it is also important to understand whether a new type has been generated.

### CRedit authorship contribution statement

**Eun-Su Kim:** Methodology, Validation, Visualization, Writing – original draft. **Dong-Hee Lee:** Conceptualization, Project administration, Writing – review & editing. **Gi-Jeong Seo:** Data curation. **Duck-Bong Kim:** Supervision. **Seung-Jun Shin:** Supervision.

### Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

### Data Availability

Data will be made available on request.

### Acknowledgement

This research was supported by the Ministry of Science, ICT (MSIT), Korea, under the High-Potential Individuals Global Training Program (No. 2020-0-01539) supervised by the Institute for Information & Communications Technology Planning & Evaluation (IITP). This research was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. NRF-2022R1C1C1011743). This material is based upon work supported by the National Science Foundation (NSF) under Grant No. 2015693. (contribution: IITP 50%, NRF 25%, NSF 25%).

### Appendix A. Supporting information

Supplementary data associated with this article can be found in the online version at [doi:10.1016/j.sna.2023.114205](https://doi.org/10.1016/j.sna.2023.114205).

### References

- [1] H. Walser, J.A. Shields, Traditional and emerging applications of molybdenum metal and its alloys, *International Molybdenum Association IMOA News*. 1 2007 16.
- [2] S.W. Williams, F. Martina, A.C. Addison, J. Ding, G. Pardal, P. Colegrove, Wire+ arc additive manufacturing, *Mater. Sci. Technol.* 32 (2016) 641–647.
- [3] G. Marinelli, F. Martina, S. Ganguly, S. Williams, Development of wire + arc additive manufacture for the production of large-scale unalloyed tungsten components, *Int J. Refract Met. Hard Mater.* 82 (2019) 329–335, <https://doi.org/10.1016/j.jrmhm.2019.05.009>.
- [4] S. Islam, G.-J. Seo, Md.R.U. Ahsan, H. Villarraga-Gómez, H.-J. Lee, D.B. Kim, Investigation of microstructures, defects, and mechanical properties of titanium-zirconium-molybdenum alloy manufactured by wire arc additive manufacturing, *Int J. Refract Met. Hard Mater.* 110 (2023), 106042, <https://doi.org/10.1016/j.jrmhm.2022.106042>.
- [5] G. Tapia, A. Elwany, A review on process monitoring and control in metal-based additive manufacturing, *J. Manuf. Sci. Eng.* 136 (2014).
- [6] P.K. Rao, J. Liu, D. Roberson, Z. Kong, C. Williams, Online real-time quality monitoring in additive manufacturing processes using heterogeneous sensors, *J. Manuf. Sci. Eng. Trans. ASME* 137 (2015), <https://doi.org/10.1115/1.4029823/374977>.
- [7] F.W. Baumann, A. Sekulla, M. Hassler, B. Himpel, M. Pfeil, Trends of machine learning in additive manufacturing, *Int. J. Rapid Manuf.* 7 (2018) 310, <https://doi.org/10.1504/IJRAPIDM.2018.095788>.
- [8] M. Mani, B.M. Lane, M. Alkan Donmez, S.C. Feng, S.P. Moylan, A review on measurement science needs for real-time control of additive manufacturing metal

- powder bed fusion processes) A review on measurement science needs for real-time control of additive manufacturing metal powder bed fusion processes, *Int. J. Prod. Res.* 55 (2017) 1400–1418, <https://doi.org/10.1080/00207543.2016.1223378>.
- [9] R.P. Garcia, T. Luz, S.C. Canobre, L. Fengde, L. Xingran, L. Youzhi, A. Mazlan, H. Daniyal, A.I. Mohamed, M. Ishak, A.A. Hadi, Monitoring the quality of welding based on welding current and ste analysis, *IOP Conf. Ser. Mater. Sci. Eng.* 257 (2017), 012043, <https://doi.org/10.1088/1757-899X/257/1/012043>.
- [10] T. Hauser, A. Da Silva, R.T. Reisch, J. Volpp, T. Kamps, A.F.H. Kaplan, Fluctuation effects in wire arc additive manufacturing of aluminium analysed by high-speed imaging, *J. Manuf. Process.* 56 (2020) 1088–1098, <https://doi.org/10.1016/j.jmapro.2020.05.030>.
- [11] J. Wang, Y. Cui, C. Liu, Z. Li, Q. Wu, D. Fang, Understanding internal defects in Mo fabricated by wire arc additive manufacturing through 3D computed tomography, *J. Alloy. Compd.* 840 (2020), 155753.
- [12] Z. Pan, D. Ding, B. Wu, D. Cuiuri, H. Li, J. Norrish, Arc Weld. Process. Addit. Manuf.: A Rev. 2018 3 24 doi: 10.1007/978-981-10-5355-9\_1/FIGURES/15.
- [13] N. Li, D. Fan, J. Huang, S. Yu, W. Yuan, M. Han, Self-adaptive control system for additive manufacturing using double electrode micro plasma arc welding, *Chin. J. Mech. Eng.* 34 (2021) 1–14, <https://doi.org/10.1186/S10033-021-00581-4/FIGURES/22>.
- [14] S. Kou, *Weld. Metall.*, N. Jersey, USA 431 (2003) 223–225.
- [15] C.R. Cunningham, J.M. Flynn, A. Shokrani, V. Dhokia, S.T. Newman, Invited review article: strategies and processes for high quality wire arc additive manufacturing, *Addit. Manuf.* 22 (2018) 672–686, <https://doi.org/10.1016/j.addma.2018.06.020>.
- [16] B. Mannion, J. Heinzman, Plasma arc welding brings better control, *Tooling Prod.* 5 (1999) 29–30.
- [17] J.N. Pires, A. Loureiro, G. Bölsjö, *Welding Robots: Technology, System Issues and Application*, Springer Science & Business Media, 2006.
- [18] X. Li, Xiaodong Jia, Q. Yang, J. Lee, Quality analysis in metal additive manufacturing with deep learning, *J. Intell. Manuf.* 31 (2020) 2003–2017, <https://doi.org/10.1007/s10845-020-01549-2>.
- [19] M. Grasso, F. Gallina, B.M. Colosimo, Data fusion methods for statistical process monitoring and quality characterization in metal additive manufacturing, *Procedia CIRP* 75 (2018) 103–107.
- [20] W. Ren, G. Wen, Z. Zhang, J. Mazumder, Quality monitoring in additive manufacturing using emission spectroscopy and unsupervised deep learning, *Mater. Manuf. Process.* (2021) 1–8, <https://doi.org/10.1080/10426914.2021.1906891>.
- [21] J. Guo, A. Huang, R. Hu, H. Xu, G. Yang, S. Pang, An in-situ monitoring system for electron beam wire-feed additive manufacturing, *Sens. Actuators A* 307 (2020), 111983, <https://doi.org/10.1016/j.sna.2020.111983>.
- [22] Z. Yang, Y. Lu, H. Yeung, S. Krishnamurthy, Investigation of deep learning for real-time melt pool classification in additive manufacturing, 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), IEEE, 2019, pp. 640–647.
- [23] D. Baciou, G. Melton, M. Papaelias, R. Shaw, Automated defect classification of SS304 TIG welding process using visible spectrum camera and machine learning, (2019). <https://doi.org/10.1016/j.ndteint.2019.102139>.
- [24] A. Caggiano, J. Zhang, V. Alfieri, F. Caiazzo, R. Gao, R. Teti, Machine learning-based image processing for on-line defect recognition in additive manufacturing, *CIRP Ann.* 68 (2019) 451–454, <https://doi.org/10.1016/j.cirp.2019.03.021>.
- [25] M.R.U. Ahsan, G.-J. Seo, X. Fan, P.K. Liaw, S. Motaman, C. Haase, D.B. Kim, Effects of process parameters on bead shape, microstructure, and mechanical properties in wire+ arc additive manufacturing of Al0.1CoCrFeNi high-entropy alloy, *J. Manuf. Process* 68 (2021) 1314–1327.
- [26] H.W. Cho, S.J. Shin, G.J. Seo, D.B. Kim, D.H. Lee, Real-time anomaly detection using convolutional neural network in wire arc additive manufacturing: molybdenum material, *J. Mater. Process. Technol.* 302 (2022), 117495, <https://doi.org/10.1016/j.jmatprotec.2022.117495>.
- [27] T. Nakazawa, D. v Kulkarni, Anomaly detection and segmentation for wafer defect patterns using deep convolutional encoder-decoder neural network architectures in semiconductor manufacturing, *IEEE Trans. Semicond. Manuf.* 32 (2019) 250–256, <https://doi.org/10.1109/TSM.2019.2897690>.



Eun-Su Kim is a researcher at SK hynix in Korea. She received her master's degree at Hanyang University in Korea. Her research interests include data mining, data science, artificial intelligence, machine learning, and statistical analysis methods.





Dong-Hee Lee is an associate professor at the Department of Industrial Engineering at Sungkyunkwan University (SKKU) in Korea. He received his bachelor's degree (2006) and his PhD degree (2011) at the Department of Industrial and Management Engineering at POSTECH. Before joining SKKU, he was a senior researcher at the semiconductor division at Samsung Electronics. His current research interests include quality engineering, smart factory, big data analytics in manufacturing, design of experiments, response surface methods, and statistical process control.



Duck-Bong Kim has been working as an associate professor at the Department of Manufacturing and Engineering Technology at the Tennessee Technological University since August 2016. Before joining TTU, he was a guest researcher at the National Institute of Standards and Technology (NIST) from October 2011 to July 2016. With an MS and PhD from the School of Information and Mechatronics at Gwangju Institute of Science and Technology (GIST), Korea, Dr. Kim is a cross-disciplinary research scientist with knowledge and experience in advanced design and manufacturing engineering, including additive manufacturing (3D Printing), sustainable manufacturing, smart manufacturing, data analytics, and optical engineering.



Gi-Jeong Seo is a post-doctoral researcher at the Department of Manufacturing and Engineering Technology at the Tennessee Technological University. He received his bachelor's, master's, and PhD degrees in Metal Engineering at Pukyong National University in Korea. His research interests include advanced design and manufacturing engineering viz. additive manufacturing (3D Printing), sustainable manufacturing, smart manufacturing, data analytics, and optical engineering.



Seung-Jun Shin is an associate professor in the School of Interdisciplinary Industrial Studies at Hanyang University. He received his bachelor's degree (2002) in Mechanical Engineering at Korea University and his master's (2005) and PhD degrees (2010) at the Department of Industrial and Management Engineering at POSTECH. His current research interests include cyber-physical production systems, big data analytics in manufacturing, environmentally-conscious manufacturing, and industrial data interoperability.