energies

MDPI

Article

Features Recognition from Piping and Instrumentation Diagrams in Image Format Using a Deep Learning Network

Eun-Seop Yu [1], Jae-Min Cha [1], Taekyong Lee [1], Jinil Kim [1] and Duhwan Mun [2,*]

[1] Plant Engineering Center, Institute for Advanced Engineering, Yongin-si 17180, Korea; yes89929@iae.re.kr (E.-S.Y.); jmcha@iae.re.kr (J.-M.C.); TKLee@iae.re.kr (T.L.); jikim@iae.re.kr (J.K.)

[2] Department of Precision Mechanical Engineering, Kyungpook National University, Sangju-si 37224, Korea

* Correspondence: dhmun@knu.ac.kr; Tel.: +82-54-530-1271; Fax: +82-54-530-1278

Received: 8 October 2019; Accepted: 19 November 2019; Published: 21 November 2019

Abstract: A piping and instrumentation diagram (P&ID) is a key drawing widely used in the energy industry. In a digital P&ID, all included objects are classified and made amenable to computerized data management. However, despite being widespread, a large number of P&IDs in the image format still in use throughout the process (plant design, procurement, construction, and commissioning) are hampered by difficulties associated with contractual relationships and software systems. In this study, we propose a method that uses deep learning techniques to recognize and extract important information from the objects in the image-format P&IDs. We define the training data structure required for developing a deep learning model for the P&ID recognition. The proposed method consists of preprocessing and recognition stages. In the preprocessing stage, diagram alignment, outer border removal, and title box removal are performed. In the recognition stage, symbols, characters, lines, and tables are detected. The objects for recognition are symbols, characters, lines, and tables in P&ID drawings. A new deep learning model for symbol detection is defined using AlexNet. We also employ the connectionist text proposal network (CTPN) for character detection, and traditional image processing techniques for P&ID line and table detection. In the experiments where two test P&IDs were recognized according to the proposed method, recognition accuracies for symbol, characters, and lines were found to be 91.6%, 83.1%, and 90.6% on average, respectively.

Keywords: deep learning; piping and instrumentation diagram; object recognition

1. Introduction

A piping and instrumentation diagram (P&ID) is a key drawing widely used in the energy industry including petroleum and power plants. P&ID is drawn based on a process flow diagram (PFD), which is a detailed schematic representation of the general flow of major equipment and materials involved in each plant process and the working fluid. Thus, a P&ID provides the basic plant design and serves as a basic resource for detail design, procurement, construction, and commissioning of a plant.

In a digital P&ID, all objects drawn are classified and made amenable to computerized data management. Symbols are a critical component of any digital P&ID and are largely categorized into four types: fitting, instrumentation, equipment, and diagram reference. Digital P&ID symbols are connected by lines. Apart from these diagram symbols, a digital P&ID contains components such as outer borders, title boxes, characters, and tables. A digital P&ID is assigned a tag ID, which is one of the attributes of a symbol, and, thus, can be linked to the project database including 3D design models. For this networking capability, a digital P&ID at industrial sites is also called an intelligent or smart P&ID.

A digital P&ID is used by most engineering, procurement, and construction (EPC) contractors responsible for plant design, procurement, and construction. However, new plants often use

image-format P&IDs that are drawn in the front-end engineering and design (FEED) stage or produced by equipment and materials manufacturers. However, plant operators use digital P&IDs and archive them as image files and large volumes of P&IDs in old plants.

To overcome these issues, P&IDs need to be converted from an image to a digital format in the energy industry. In this case, digitalization is a process of recognizing high-level objects with field-specific meanings among diagram images and extracting necessary information from them, which is followed by replicating the original diagram images. Currently, the P&ID process is mostly manual and its quality varies depending on the skills of the individual worker, which makes it a time-consuming and error-prone undertaking.

To generate a digital P&ID from a diagram, a method to recognize and extract each individual object contained in the P&ID drawings should be developed. In this paper, a deep learning-based method to recognize and extract critical information contained in P&ID drawings is proposed. The method consists of two stages—preprocessing and recognition. The preprocessing stage is further divided into process steps such as diagram alignment and removal of outer borders and title boxes. In the recognition stage, symbols, characters, lines, and tables are recognized. A deep learning model is used for symbol and character recognition. To develop the deep learning model for P&ID recognition, its training data structure must be properly defined.

The rest of this paper is organized as follows. Section 2 analyzes previous research on diagram recognition. Section 3 analyzes the P&ID structure, identifies information necessary for object recognition, and defines the deep learning training data structure. Section 4 presents the preprocessing algorithm conducted prior to the P&ID object recognition stage. Section 5 presents the P&ID object recognition algorithm. Section 6 discusses recognition experiments of symbols, characters, and lines with two-test P&IDs. Lastly, conclusions are drawn in Section 7.

2. Brief Literature Review

Major components of a diagram include symbols carrying field-specific meanings, lines representing inter-symbol connections, and attributes assigned to symbols and lines through texts or characters [1]. There have been studies on the recognition of diagrams such as electric diagram [2,3], engineering diagram [4–6], and logic diagram [7]. These research studies used traditional image recognition methods using geometric features of objects. After recognizing the discernible features from analog diagrams using techniques such as edge detection [8], Hough transform [9], morphological operation [10], and image feature extraction [11], symbols and lines within the image are searched by comparing the features with those extracted from the predefined symbols and lines. These traditional recognition methods reach their limitations when conducting robust recognition of various types of objects (symbols, lines, texts, etc.) contained in analog diagrams. In addition, accuracy of object recognition is compromised in the presence of inter-object interruptions, morphological changes, or noises such as stains.

Some studies on recognizing P&IDs have also been conducted recently [4,6,12,13]. To recognize symbols, lines, and texts in P&IDs, these research studies also applied traditional image recognition methods. Specifically, Reference [4] proposed a feature-based method recognizing symbols, lines, and texts of a plant diagram, which is almost the same as P&IDs [12]. Tan et al. (2016) proposed a new method based on branches of image feature extraction called local binary pattern (LBP) and spatial pyramid matching (SPM) to recognize symbols and texts in a P&ID. Arroyo et al. (2016) recognized symbols and texts using Hough transformation, generalized Hough transformation, and optical character recognition (OCR). The recognized information was converted into a plant simulation model [13]. These studies had to use simplified P&IDs since the performance of traditional feature recognition methods highly vary with changes in conditions such as object rotations, flips, scaling, and noises. Likewise, due to technological difficulties, the recognition of texts in P&IDs have also been conducted restrictively with rule-based approaches. Recently, more advanced research on recognizing complex P&IDs has been conducted using various feature-based methods and Tesseract OCR engine [6]. However, it is still difficult to overcome the inherent limitations of the traditional methods.

Meanwhile, a convolutional neural network (CNN) [14], which is a class of deep neural networks optimized for analyzing visual imagery, has emerged as an alternative way to overcome technological limitations of the traditional image recognition methods. As a deep learning model for classifying image classes or categories, GoogLeNet, which won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2014 [15] with the top-5 error rate at 6.67% in images classification of 1000 categories, surpasses human-level performance in object recognition and classification. You only look once (YOLO) [16], region-CNN (R-CNN) [17], and single shot detector (SSD) [18] are examples of CNN-based deep learning models that detect the position of a specific class of a specific object within the range of an overall image. Other image-classification deep learning models include image detection methods employing a sliding window [19] or a fully convolutional layer [20]. There are a variety of open-source tools such as TensorFlow [21], Keras [22], Pytorch [23], and Theano [24], and the popularization of parallel processing using graphics processing units (GPUs) has accelerated the spread of deep learning models in various fields including image processing.

Through enough training on data, CNN can overcome the limitations of the conventional methods such as inter-object interruption, morphological change, and noise problems. This feature makes CNN much more effective than the conventional methods in object recognition. The conventional methods require detailed calibrations of pre-defined templates when changes such as rotations, interruptions, and scaling are applied to target objects. However, if given enough data and computing resources, CNN can deal with those unexpected conditional changes by itself. This is because CNN calculates an object's features with convolutional layers and sorts out only the features that distinguish each object while input data is going through multiple layers of CNN [25].

To leverage the advances in deep learning techniques, several researchers have employed CNN-based deep learning techniques in recognition-related studies [26,27]. However, Reference [26] is a simple logic diagram for general use and, although it deals with the recognition of P&ID objects, Reference [27] considers neither the stage of diagram preprocessing nor line and table recognition. In contrast, the method proposed in this paper recognizes not only symbols and lines, which are key components of P&ID objects, but also texts and tables. This study differentiates itself from previous studies in that it improves the diagram recognition efficiency by providing a preprocessing stage. In addition, a training data structure for P&ID object recognition is also proposed.

3. Training Data Definition for P&ID Recognition

In ISO 10628 [28], P&ID is defined as a diagram that, based on the process flow diagram, represents the technical realization of a process by means of graphical symbols for equipment and piping together with graphical symbols for process measurement and control functions. Most of the deep learning models for image detection and classification with high accuracy are trained with common object data such as human beings, animals, plants, and cars. Therefore, they cannot be directly applied for P&ID object recognition, and should be upgraded with additional training data including symbols, characters, and text data used in P&IDs. It is crucial to secure a sufficient amount of training data in order to improve the P&ID recognition accuracy using a deep learning model. The training data structure suitable for P&ID recognition needs to be defined in this sense to ensure the efficiency of training data building by collecting all necessary data and consistently extracting relevant data from all pertinent sources [29].

3.1. Training Data Requirement Analysis for P&ID Recognition

A P&ID is composed of outer borders and title boxes, as well as symbols, lines, characters, and tables. This study defines symbols, texts, lines, and tables as the objects to be recognized. Key information elements for symbol recognition using a deep learning model are class, bounding box, and orientation. A bounding box is expressed in terms of the coordinates of the top-left and bottom-right corners, and orientation is expressed in terms of the four cardinal directions rather than angles. Key information elements for character recognition are character count, bounding box, and orientation.

Key information elements for line recognition are class, start point, end point, the line, point, edge morphologies, and the position orientation of the flow sign. Line, point, and edge morphologies indicate whether the starting and ending points of a line are connected via an arrow. The flow sign indicates the fluid flow direction along the pipe. Its position is expressed by the coordinates at the top-left and bottom-right corners of the bounding box surrounding it. The orientation of the flow sign is expressed in terms of the four cardinal directions.

3.2. Training Data Structure Definition for P&ID Recognition

Figure 1 illustrates the training data structure used for the P&ID recognition deep learning model. The P&ID folder is the layer on top of the folder. It consists of the files containing the lists of symbol and character classes. The symbol class includes the symbol type and orientation as well as character class, count, size, and orientation. Projector folders are placed under the P&ID folder layer. Training data are generally stored at the project unit because each project uses its own symbols and legend. A project folder, thus, stores files containing the list of the diagrams created in that project and actual image format diagrams.

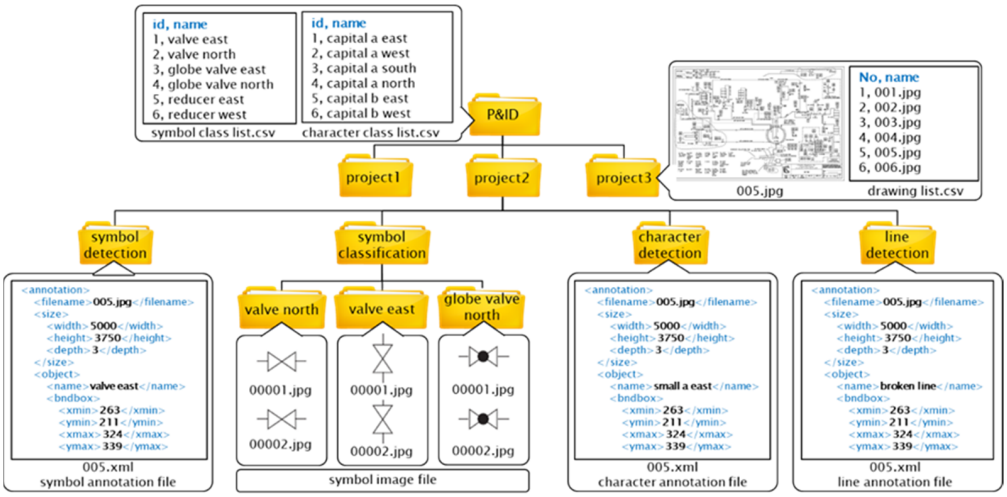


Figure 1. Training data structure used for P&IDs object recognition.

In the project folder layer, there are four training data folders assorted by task area including symbol detection, symbol classification, character detection, and line detection. In the training data folders for character detection and line detection, each diagram has an annotation file with data needed to meet the requirements explained in Section 3.1.

For the definition of the annotation file data structure, we benchmarked the data structure used at the Visual Object Classes (VOC) challenge held by Pattern Analysis, Statistical Modelling, and Computational Learning (PASCAL) [30]. In the training data folder for symbol classification, symbols cut out from the diagram are stored by class because symbol images themselves are training data.

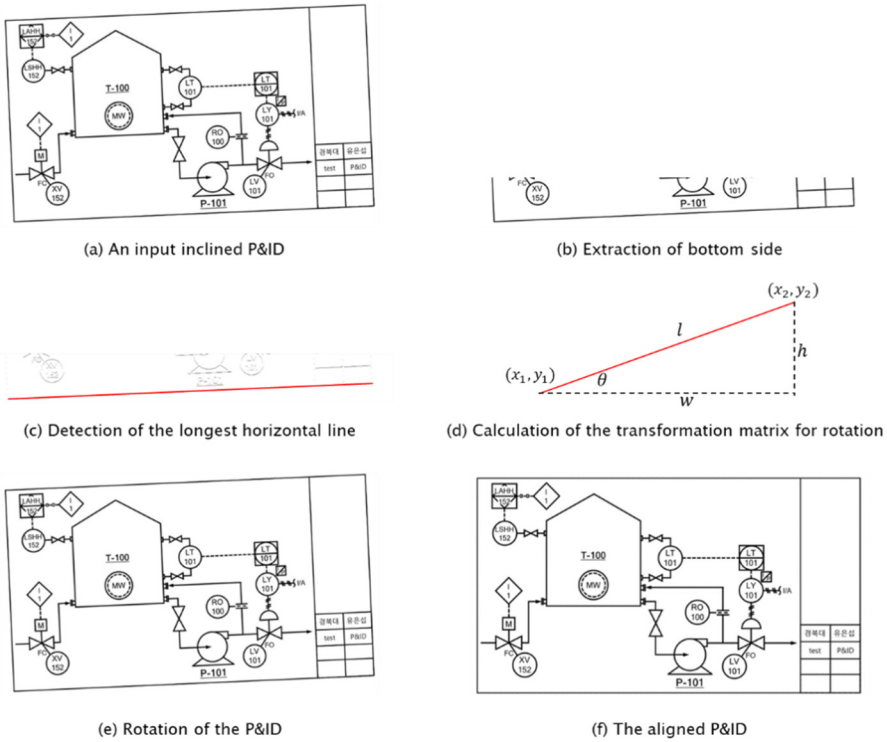
4. P&ID Recognition Accuracy Improvement through Preprocessing

4.1. Diagram Alignment

Prior to the popularization of digital P&IDs, most diagrams were drawn on paper or as image files, whereby the latter was generated by scanning hand-drawn diagrams. When the diagrams are scanned in an inclined state, depending on the environment or condition, more training data will be required to process various inclined angles of the scanned images. In the case where the amount of training data is limited, preprocessing must be conducted to realign the inclined diagram, in order to improve the diagram recognition accuracy. In other words, an inclined diagram must be realigned in the preprocessing stage to improve the diagram recognition accuracy using a limited amount of training data.

To realign the inclined P&IDs (Figure 2a), the degree of inclination or tilt angle must be measured. Given that the longer the straight line used for the angle calculation, the smaller the angle calculation error is, the horizontal line of the outer border, which is the longest line of a diagram at the outer boundary, is used for calculating the rotation angle. To identify the outer border horizontal line, the bottom one-fifth of the diagram is cut out as shown in Figure 2b, in which the longest horizontal line is identified as shown in Figure 2c. A detailed method for horizontal line recognition is described in Section 5.3. The inclined angle, as shown in Figure 2d, can be calculated using an arc-sine function after obtaining the coordinates of both ends of the longest horizontal line of the diagram. After calculating its inclined angle, the diagram is realigned by image rotation, as shown in Figure 2e. The inclined angle θ is calculated using Equation (1).

$$\theta = \sin^{-1}(y_2 - y_1) / \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \quad (1)$$



(a) An input inclined P&ID (b) Extraction of bottom side

(c) Detection of the longest horizontal line (d) Calculation of the transformation matrix for rotation

(e) Rotation of the P&ID (f) The aligned P&ID

Figure 2. Alignment of an inclined P&ID.

4.2. Removal of Outer Borders and Title Boxes

Preprocessing for P&ID recognition includes the outer border and title removal, as well as diagram realignment. If the outer borders and the title boxes are included in a diagram, they will affect the character and table recognition process whereas no significant morphological differences will be observed in symbols, lines, and tables. Accordingly, to enhance the accuracy of the diagram object recognition, it is necessary to remove them in the preprocessing stage itself.

The morphologies of the outer borders and the title boxes in a P&ID may vary from project to project. An outer border is marked by a single or double solid line, often lined with minor ticks. A title box is placed in the bottom left, center, or right. Annotations inserted in a P&ID may be in a note area usually placed at the right corner of the diagram.

Figure 3 illustrates the border and title box removal process. After cutting out the bottom one-fifth of a diagram (Figure 3b), all horizontal lines on this part of the diagram are clipped out. Among the extracted horizontal lines, the longest line is selected, and the second longest line is also selected if the length error is within a 5% range (Figure 3c). Once a horizontal line is selected, all lines connected

to this line are identified by detecting all the black pixels on the diagram image connected to the line (Figure 3d). Since a title box is connected to the outer border, the title box can be identified by extracting the black pixels on the image corresponding to the outer border. A morphology operation is then applied to the extracted black pixels to identify the outer border and title box (Figure 3e). Lastly, the identified areas are removed from the P&ID image (Figure 3f).

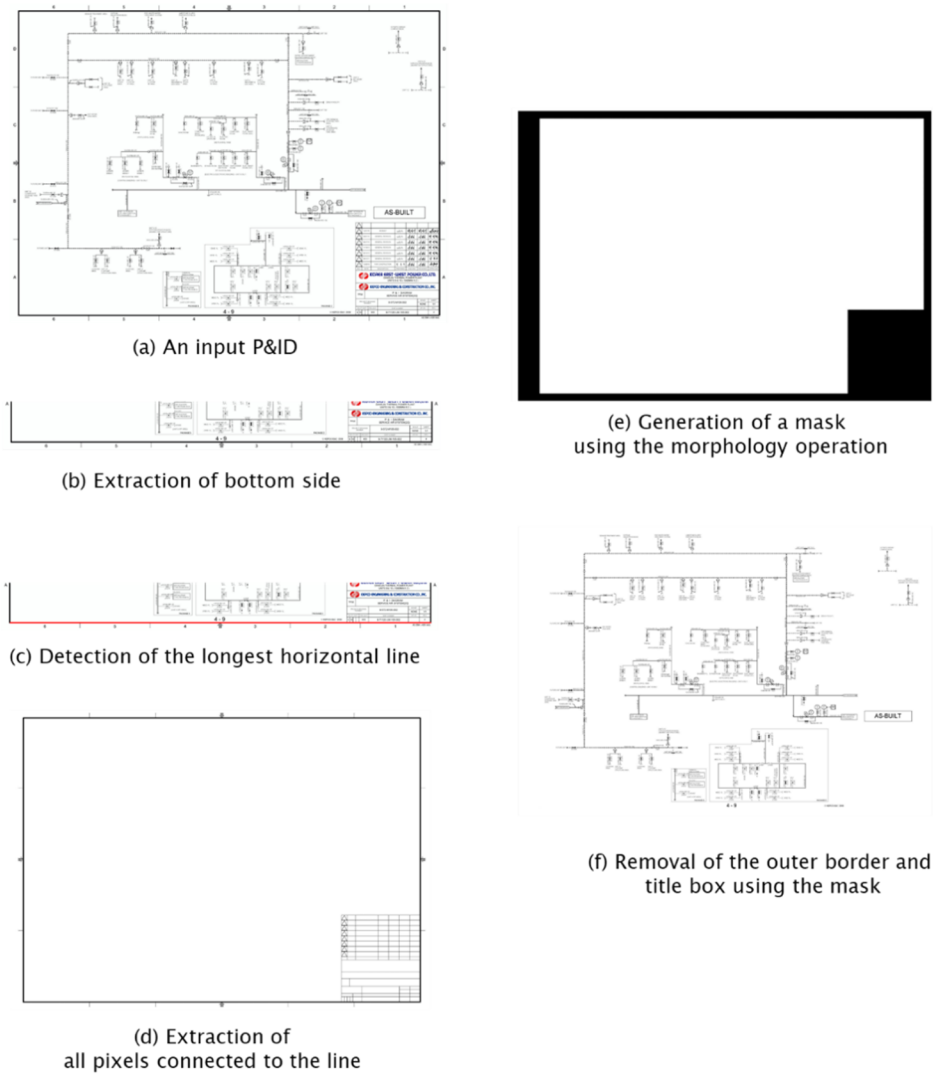


Figure 3. Removal of the outer border and title box in P&ID.

Morphology operation is a technique commonly used for image preprocessing or postprocessing. The common basic morphology operations include erosion and dilation operations, in which the respective minimum and maximum values of the pixels in the kernel area are assigned to the value of the current pixel. There are also open and close operations, in which the basic operations are combined. The combination has the effect of removing minor areas and filling empty spaces, respectively, while retaining the overall morphology. We used the open operation to extract the outer borders and title boxes.

5. Object Recognition on an Image-Format P&ID

5.1. Symbol Detection

Before proceeding to symbol recognition, the image-classification deep learning model learns how to classify images with the symbol image training data extracted from the diagram. It then detects the types and positions (bounding box) of the symbols included in the P&ID image, by applying the sliding

window technique to the trained deep learning model. Lastly, the overlapping symbol recognition results are merged through the process of grouping.

A new deep learning model for symbol image classification is defined based on AlexNet [14]. AlexNet is a CNN model that won the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) for classifying an image with 1000 classes. This model consists of eight layers, and its recognition accuracy is lower than those of GoogLeNet [15], VGGNet [31], and ResNet [32], which have more than 20 layers. However, considering the limited volume of data in the engineering sector compared with other fields where thousands of pages of training data are available, we defined the deep learning model for symbol image classification based on AlexNet, which can be trained with a relatively small amount of data.

The convolutional layer calculates the output value for each input image at each output node over the entire image pixel area, using a convolutional filter of a specific size, and transmit the value to the next neuron. The convolutional layer's parameters are the filter size, number, stride, padding, and activation function. Filter size is the size of filter running through the input image pixels. The filter number is the number of filters used per one image. Stride is the distance covered by the filter to move from pixel to pixel in the image. Padding is the range of image expansion in the four cardinal directions by the filter prior to image scanning. The padding value in image recognition is generally set as the "same," which means an input data size expansion such that the data size after passing the convolutional layer remains the same as before passing it. An activation function is the function that delivers an input value of the neuron surpassing the reference value to the next neuron. A fully connected layer is the layer connected to all neurons in an adjacent layer. The parameters of a fully connected layer are the number of output data values and the activation function.

Figure 4 presents the structure of the image classification's deep learning model defined in this study. Like AlexNet, this model has five convolutional layers and three fully connected layers. However, the training image used in AlexNet is a three-channel red, green, blue (RGB) image. One color channel was sufficient for our model because the P&ID drawing used in this study was a black and white image. Thus, the size of input data was set to $227 \times 227 \times 1$. The parameters of the first CNN were filter count (32), size (11), stride (4×4), and padding (same). After the first CNN layer, a pooling layer (size: 3×3, stride: 2×2) was applied. The parameters of the second CNN layer were filter count (86), size (5×5), stride (1×1), and padding (same), to which the same pooling layer as in the first CNN layer was applied. The parameters of the third and fourth CNN layers were filter count (128), size (3×3), stride (1×1), and padding (same), and no pooling was applied. The parameters of the last CNN layer were filter count (86), size (3×3), stride (2×2), and padding (same), to which the same pooling layer as in the first CNN layer was applied. The first and second fully connected layers consisted of the number of output data (1364), to which 1/2 dropout was applied. The parameter of the third fully connected layer—the output layer—was the number of output data (9), which is the number of classes of the image to be recognized. Using softmax as the activation function in the last fully-connected layer, the probability that the input image would correspond to nine classes was derived. ReLU was used as the activation function of the CNN layers and the fully connected layers except for the output layer. The image-classification deep-learning model training was implemented using softmax as the activation function of the last layer. However, post-training prediction using the model was performed without softmax in the last layer, to improve the operation speed.

For this study, four types of symbols were selected: pump, instrument, valve, and diagram reference (DR). Considering the possible symbol alignment orientations on the diagram, the recognition objects were categorized into nine classes: pump, DR I (inlet) east, DR I west, DR O (outlet) east, DR O west, instrument north, instrument west, valve north, and valve east. The reason for selecting these particular types of symbols is the largely varying morphology, size, and aspect ratio of each of these symbols and the importance of their accurate recognition for the success of the proposed deep learning model through training data expansion.

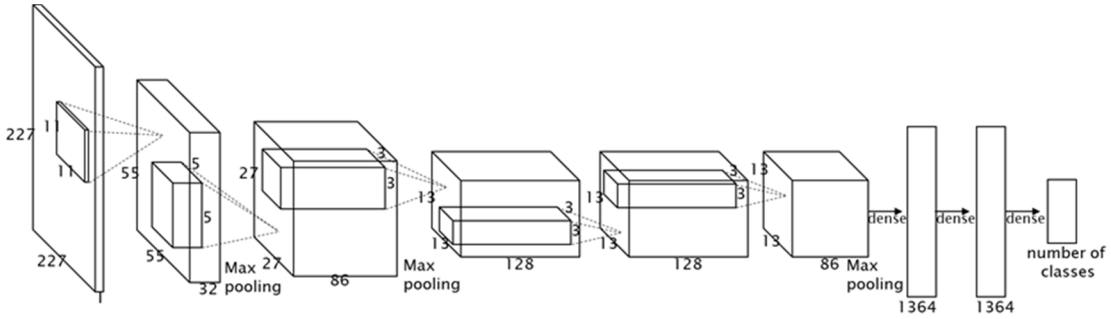


Figure 4. Deep learning model used for the classification of symbol images.

In general, there is a large variety and amount of training data available for general usage, such as the Modified National Institute of Standards and Technology (MNIST) [33] and the Canadian Institute for Advanced Research (CIFAR) [34]. However, there are hardly any open-access training data in specialized fields such as P&ID. In addition, there has not been any significant research on data extraction. Therefore, in this study, some of the P&ID symbols were set as recognition objects, and the image and class of the symbols selected for training were matched manually. To construct the training data, we extracted 28 symbol images for each class from 68 P&IDs provided by the 'S' engineering company in Korea. To increase the number of training data, a rotation transformation between −10° and +10° was performed on each extracted symbol image. Furthermore, a translation transformation between 0% and 5% on the size of each symbol image along the x and y directions was also performed, followed by the conversion of each symbol image to 227 × 227. It yielded 140 symbol images per class, of which 100 were used for training and the remaining 40 were used for validation, as shown in Figure 5.

Class (label)	Symbol images	Amount
instrument north	XV 10091 PI 10021 TI 10021 FQI 10021 PI 10091 TI 10091 FQI 10091 FQI 10031 PI 10022	140
instrument west	PI 30066 XV 30065 XV 30066 XZV 30071 XZV 30070 XV 30074 XV 30079 PIC 30079 XV 30072	140
DR I east	PID0106 PID0203 PID0202 PID0410	140
DR I west	PID0104 PID0214 PID0606 PID0401	140
DR O east	PID0103 PID0103 PID0215 PID0101	140
DR O west	PID0305 PID0303 PID0305 PID0304	140
pump		140
valve north		140
valve east		140

Figure 5. Training data used for the recognition of symbols with the help of a deep learning model.

In the training process, cross entropy was used as the loss function. The adam optimizer, which has a learning rate of 0.0001 and a decay rate of 0.000001, was chosen as the optimizer. The batch size and epoch were set to 50 and 150, respectively, and the training data was configured to be shuffled after every epoch. The classification accuracy for the post-training verification data was 99%. Table 1 presents the prediction score by classifying each recognition symbol, whereby the prediction score is the output of the predicted value without having to use softmax in the last layer of the deep learning model.

Table 1. Prediction scores for trained symbol types.

Class (Label)		Instrument North	Instrument West	DR I East	DR I West	DR O East	DR O West	Pump	Valve North	Valve East
Prediction score	instrument north	**17.05** *	6.36	0.45	−4.54	−4.10	−4.40	0.84	−6.95	−5.3
	instrument west	−1.67	**26.48**	−10.82	3.51	−12.44	3.73	1.19	−6.71	−4.55
	DR I east	1.41	−2.32	**19.39**	−3.28	2.26	0.35	−4.96	−11.21	1.83
	DR I west	−6.16	5.35	−8.24	**19.82**	−8.32	1.68	4.08	−12.69	2.69
	DR O east	−1.50	−4.81	3.06	−1.88	**17.14**	0.99	−8.05	−2.15	−6.49
	DR O west	−6.71	−0.67	−6.97	5.70	2.33	**16.15**	−7.63	−0.97	−7.26
	pump	10.53	5.86	−3.57	−6.28	−14.71	−17.82	**31.13**	−10.78	6.83
	valve north	−0.62	−0.43	−12.55	−7.51	3.05	3.04	−2.50	**17.73**	−7.05
	valve east	−2.78	0.41	2.04	7.14	−9.81	−11.62	7.07	−13.82	**24.67**

* **XX.XX**: the highest prediction score in each row.

The results of the correct answers for predicting the symbols trained using the classification model suggests that the pump symbol scored the highest with 31.13 points, while the DR O west symbol scored the lowest with 16.15 points. When filtering the symbol prediction results according to the score, the predicted value should be normalized to ensure that its importance does not change with the symbol type. Therefore, the prediction value was normalized with Equation (2). PC_n is the normalized prediction score (PC), PC is the prediction score, and PC_m is the maximum PC in a specific class.

$$PC_n = PC/PC_m \quad (2)$$

For instance, if the prediction value for a given image to be classified as the pump class is 23, then the normalized value will be 0.74 because the maximum predicted value of the pump class is 31.13. In the in-image object detection method using a sliding window, image classification was performed, while a window with a specific size slid over all the pixels inside an image. However, an image with a high resolution, such as 5000 × 3000, inevitably takes a longer operation time. To address this problem, candidate areas likely to have symbols were identified in a high-resolution diagram, and a sliding window was applied to those areas, which reduced the amount of operation required. Especially, to identify an area containing a symbol in a P&ID drawing, a close morphological operation was performed and the empty spaces inside the symbol were filled, which was followed by an open morphological operation to remove elements such as piping. Figure 6 illustrates the results of the symbol area detection.

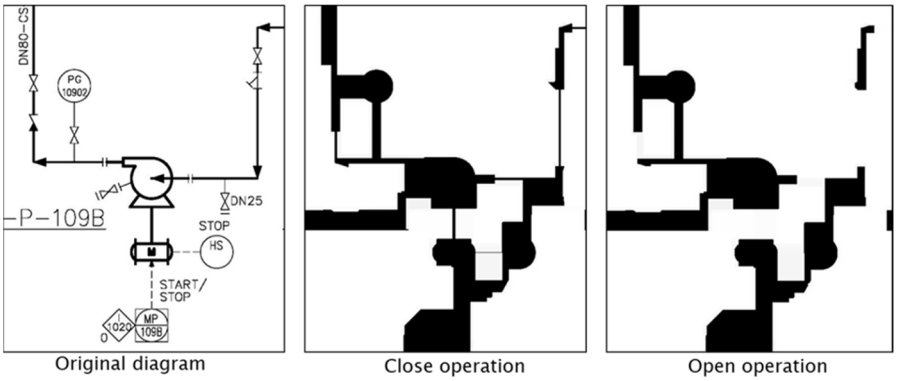


Figure 6. Identification of regions in a diagram where the symbols possibly exist.

After identifying the areas likely to contain symbols, a sliding window was applied to the identified areas. From each pixel inside that area, an image corresponding to the size of the symbol of each class was extracted. In this study, five images of different sizes were extracted at each pixel over which the sliding window passes, because the instrument west and instrument east have the same size, as do DR I east, DR I west, DR O east, and DR O west. Five prediction values could be obtained from each pixel

by classifying the extracted images with the image-classification deep learning model. These values were then normalized using Equation (1). Lastly, among the normalized prediction values, the one with the highest value was used.

Image-based object detection using a sliding window results in multiple windows (bounding boxes) with high prediction values. Pixels where a symbol is located and adjacent pixels have high prediction values as well. To ensure accurate detection of the target object from multiple identified windows, it is necessary to group the prediction results together and select the one with the highest probability of being the position of the target object. The grouping of the adjacent predicted windows is performed using intersection of union (IOU), as described in Figure 7. IOU is a metric used to evaluate how similar one predicted window (bounding box) is to another predicted window. An IOU score of 1 means that two predicted windows precisely match each other and an IOU score of 0 means that two predicted windows do not overlap at all. IOU is defined using Equation (3). A_o is the area of overlap and A_u is the area of union.

$$IOU = A_o/A_u \quad (3)$$

In the grouping process, the leftmost prediction result in the diagram was detected and grouped together with the adjacent ones. The adjacent prediction results are defined as those with an IOU greater than or equal to a predefined threshold value (i.e., 0.3). After the prediction result farthest from the first selected prediction result in the group has been selected, the process is iterated until there is no adjacent prediction result left in that group. After the grouping process of a group has been terminated, the whole process is applied to the next ungrouped prediction results and is iterated until there are no more ungrouped prediction results left.

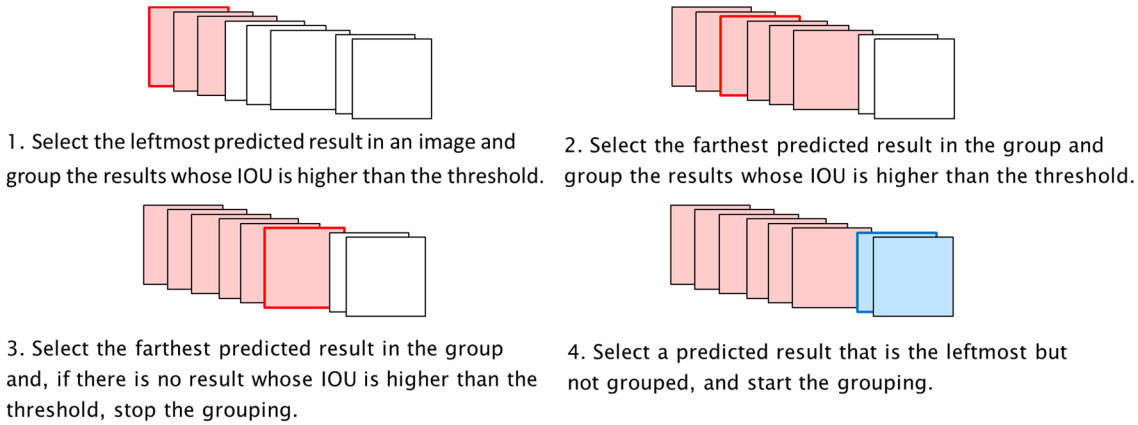


Figure 7. Grouping of adjacent predicted results.

The method of grouping the adjacent prediction results was verified by applying a sliding window to detect the pump symbol in the P&ID and performing the grouping of prediction results, as shown in Figure 8.

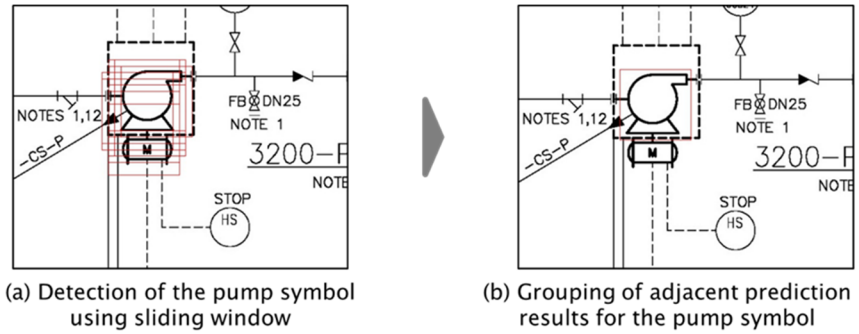


Figure 8. Grouping of predicted results for the symbol.

5.2. Character Detection

To find the character position, the connectionist text proposal network (CTPN) [35], which is a text detection deep learning model, was used. CTPN first predicts the text and non-text areas in the input image using a CNN to recognize a text area. In this process, it may take text-like patterns, such as bricks and leaves, for the text areas. To solve this problem, CTPN applies a long short-term memory network (LSTM) [36], which is a type of recurrent neural network (RNN), along with CNN, which improves the text detection accuracy of verifying whether both ends of the detected area are connected to the characters.

Data used for training CTPN include approximately 3000 images, including those provided by the International Conference on Document Analysis and Recognition (ICDAR) 2013 [37], and those collected by the CTPN developers themselves.

5.3. Line Detection

In a P&ID, lines as objects are divided continuously and dotted lines are divided by type, while horizontal, vertical, and diagonal lines are divided by orientation. In general, in a P&ID, a continuous line represents a pipeline and a dotted line represents a signal line. For line detection training, the continuous line type and the horizontal and vertical line orientations were chosen. Additionally, the lines pertaining to symbols were excluded from the recognition objects.

The P&ID line detection was performed in three steps. First, the thickness of the line most frequently used was determined and the thickness of all P&ID lines was compressed to one pixel. The coordinates and lengths of the line objects were then extracted by identifying the black pixels connected to each other in the thickness-compressed P&ID. Lastly, the lines separated in the thickness compression process were merged based on the thickness of the most frequently used line.

Figure 9 depicts the method to identify the thickness of lines frequently used in a P&ID. The number of connected black pixels was recorded while moving pixel by pixel from the leftmost one toward the right-hand side of the image. Likewise, the number of connected black pixels was recorded while moving pixel by pixel from the topmost pixel toward the bottom of the image. Lastly, the two records were combined and the distribution of the pixel count over length was calculated. Since a considerable portion of the line objects constituting the P&ID were continuous lines (pipelines) or dotted lines (signal lines), the approximate thicknesses of the continuous and dotted lines could be determined by checking the pixel length that occupied the largest proportion in the pixel length distribution.

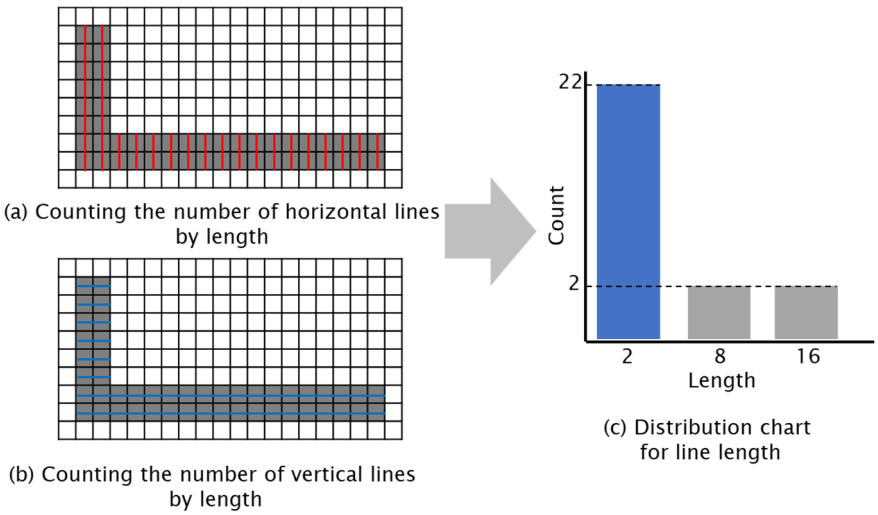


Figure 9. Generation of the distribution chart for line length.

Figure 10 illustrates the method to compress the thickness of the lines in the P&ID to one pixel. First, the kernel for line thickness compression was defined. When (x, y) is the coordinate of the target pixel, a 1 × 2 kernel makes the target pixel white if the (x + 1, y) pixel is black. By applying the 1 × 2 kernel to all pixels of a P&ID image, horizontal lines are eliminated, and an image with a line thickness of one pixel can be obtained with the remaining lines. When (x, y) is the coordinate of the target pixel, a 2 × 1 kernel makes the target pixel white if the (x, y + 1) pixel is black. By applying the 2 × 1 kernel to all pixels of a P&ID image, horizontal lines are eliminated and an image with a line thickness of one pixel can be obtained with the remaining lines.

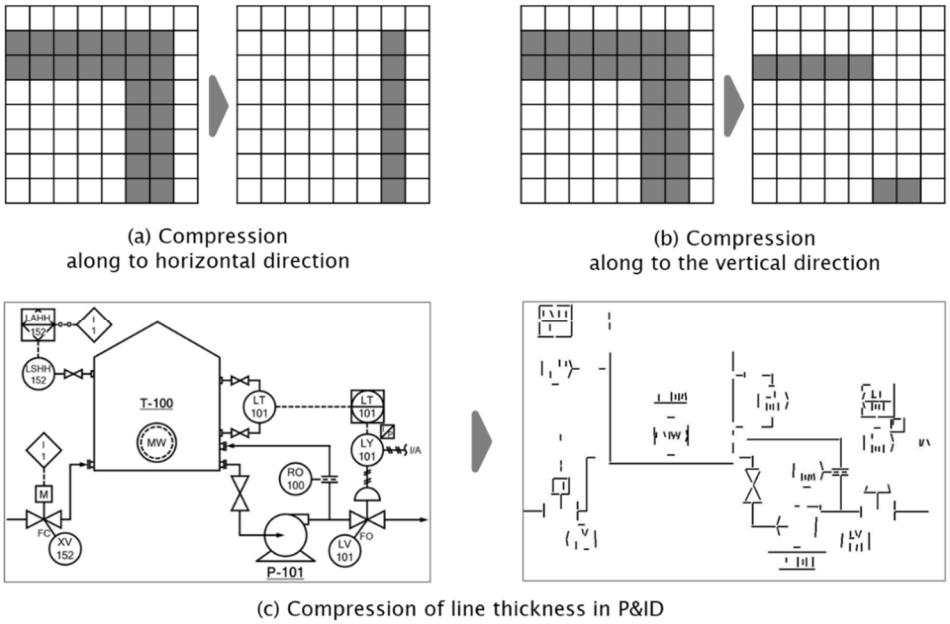


Figure 10. Compression of line thickness in P&ID.

The coordinates and length of a line object can be extracted from the compressed P&ID through the following steps. In the case of a vertical line object, black pixels are investigated by moving from the topmost pixel toward the bottom of the image obtained after the horizontal lines have been removed. If a black pixel is detected, its coordinates are recorded and the vertical line detection kernel is applied. If the current kernel position is (x, y), the vertical line detection kernel checks the values of the (x, y + 1), (x − 1, y + 1), and (x + 1, y + 1) pixels and moves to the next black pixel, repeating the process until there are no more black pixels. After the termination of the vertical line detection kernel, the starting and ending points of the line can be defined by recording the coordinates of the last pixel. In the case of a horizontal line object, black pixels are investigated while moving from the leftmost pixel toward the right-hand side of the image obtained after the vertical lines have been removed. If the current kernel position is (x, y), the horizontal line detection kernel checks the values of the (x + 1, y), (x + 1, y − 1), and (x + 1, y + 1) pixels. Other process steps are the same as in the vertical line detection method.

The method of merging the lines separated in the line thickness compression process is implemented in the following steps. The starting point of all vertical line objects detected in the previous stage is above the ending point. Likewise, the starting point of all horizontal line objects is to the left of the ending point. Therefore, the lines can be merged by comparing the gap between the ending point of the currently selected line object and the starting point of another line object, and merging the line objects whose gap is less than or equal to the reference gap. The reference gap was set to three times the thickness of the line. In the case of intersecting lines frequently appearing in the P&ID, the reference gap must be larger than the line thickness because the remaining vertical lines after the thickness compression in the horizontal direction incur a gap as thick as the horizontal line.

In consideration of the varying thickness of line objects in the P&ID, the reference gap was set to three times the thickness of the line.

As shown in Figure 10c, several lines pertaining to the symbol object were recognized line objects. Therefore, line detection accuracy can be improved by removing these misrecognized lines based on the position and size information of symbols recognized in the previous stage.

5.4. Table Detection

Some P&IDs contain tables listing detailed attributes of equipment included in the diagram, such as materials, registration, and allowable temperature and pressure. A table and the text inside it can act as noise, interfering with the recognition of symbols, lines, and texts that are the key recognition objects of the P&ID. Therefore, tables should be detected and removed. A table can widely morphologically vary. However, this study deals with the most basic form of the table in which all the lines are continuous lines. The target table has a rectangular shape made up of vertical and horizontal lines, with several vertical and horizontal lines inside it and no lines connected to outside elements.

The method to recognize a table in a P&ID is presented in Figure 11. The first step is to identify the line combinations that form a table based on the same starting and ending points of those lines when connected in the vertical and horizontal directions (Figure 11a). The second step is to generate a rectangular kernel with the identified line combinations, setting the line thickness to three times the thickness of the P&ID lines (Figure 11b) and checking whether there are lines protruding from the generated kernel and connected to other elements in the diagram (Figure 11c). Lastly, the identified line combination with no line connected to any outside element is defined as a table (Figure 11d).

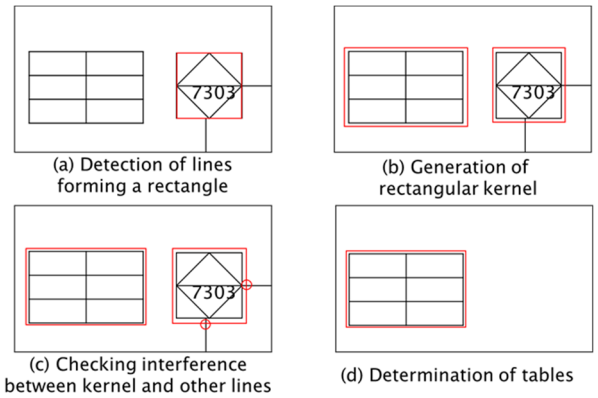


Figure 11. Table recognition in P&ID.

6. Implementation and Experiments

To prove the validity of the proposed method, a prototype system that recognizes symbols, characters, and lines from a P&ID was implemented in Python 3.7 on Windows 10 operating system. The deep learning model to recognize symbols was developed using TensorFlow 1.14.0. Computers performing the recognition of P&IDs were equipped with AMD Ryzen 7 2700X CPU, 64GB RAM, and two NVidia GeForce RTX 2080 Ti graphic cards.

Two test P&IDs are prepared for the experiments, as shown in Figure 12. They are modelled by referring to P&IDs provided by 'S' engineering company in Korea. Test P&IDs 1 and 2 have resolutions of 1460 × 840 and 1360 × 780, respectively.

Sliding window sizes during symbol recognition were set to 70 × 70 for instruments, 40 × 200 for DRs, 120 × 120 for pumps, 40 × 25 for valve north, and 25 × 40 for valve east. A normalized prediction score value used for determining whether a specific symbol exists inside a sliding window at a specific position was set to 0.8. A threshold IOU value used for grouping adjacent predicted windows was set to 0.33.

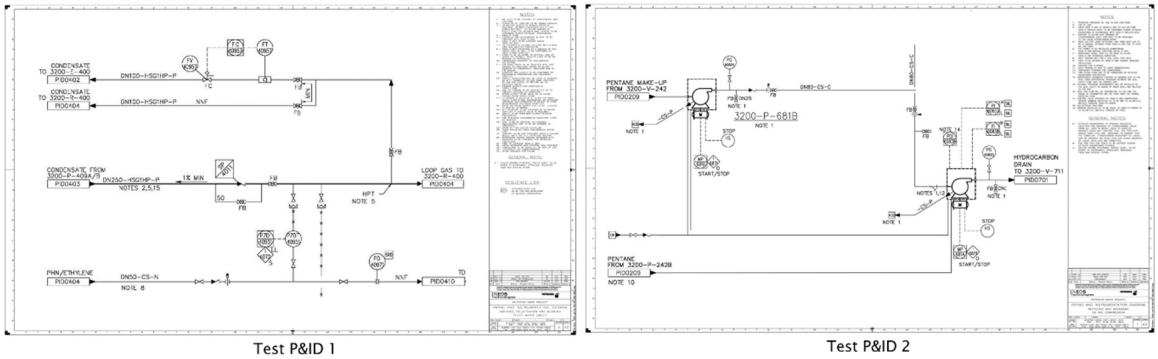


Figure 12. P&IDs used for the experiments.

Preprocessing of P&IDs and recognition of symbols, characters, and lines from P&IDs were proceeded automatically without human intervention using the prototype system. Time required to train the deep learning model used for symbol recognition was approximately 12 min. It took less than one minute to remove the outer border and title box at the preprocessing stage. It took 36.5 min, 5.53 s, and 3.25 s on average respectively to recognize symbols, characters, and lines at the recognition stage.

Recognition results are summarized in Table 2 and Figure 13. In Table 2, two metrics of recognition accuracy R and misidentification rate M were defined to measure the performance with Equations (4) and (5). NO_d is the number of objects detected, NO_s is the number of objects of a specific type in a P&ID, N_m is the number of misidentified objects, and NO_i is the number of objects identified as a specific type. Recognition accuracy indicates the ratio of detected objects in all objects of a specific type in a P&ID, and the misidentification rate indicates the ratio of misidentified objects in all objects identified as a specific type.

$$R = NO_d / NO_s \quad (4)$$

$$M = N_m / NO_i \quad (5)$$

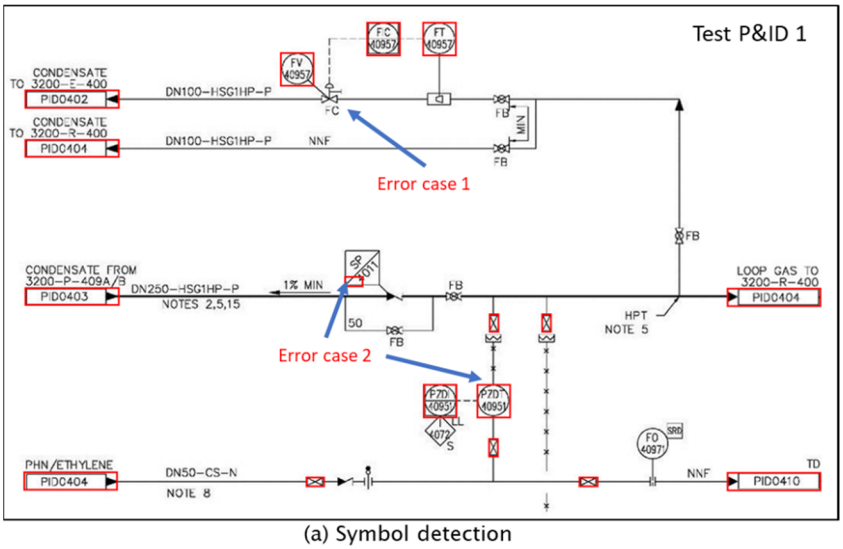
Table 2. Recognition accuracies and misidentification rates for test P&IDs.

Diagram Name	Recognition Accuracy			Misidentification Rate		
	Symbol	Character	Line	Symbol	Character	Line
Test P&ID 1	92.3%	84.2%	93.7%	25%	18.8%	17.9%
Test P&ID 2	90.9%	82%	87.5%	22.2%	19.6%	6.25%
Average	91.6%	83.1%	90.6%	23.6%	19.2%	12.1%

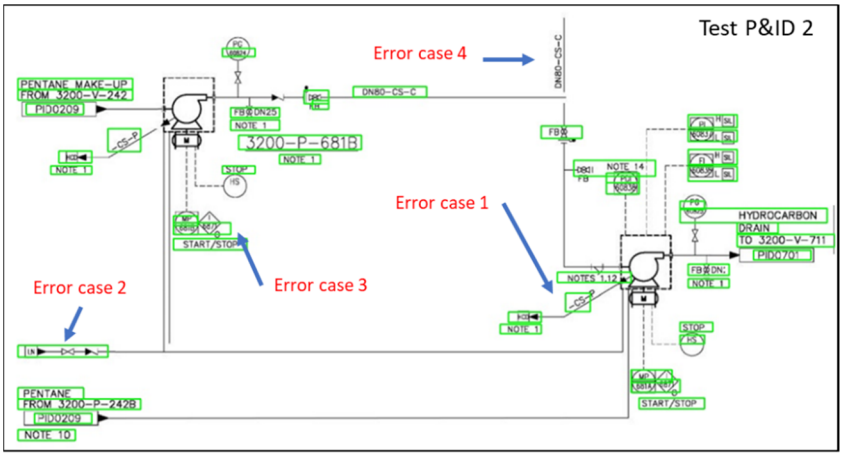
Symbol recognition accuracy was 91.6% on average. Character recognition accuracy was 83.1% on average and line recognition accuracy was 90.6% on average. Regarding the symbol recognition result, some of the target symbols were not recognized from the test P&IDs (error case 1 of Figure 13a). However, almost all targeted symbols of nine types were recognized from the test P&IDs. High recognition accuracy was achieved because a small number of symbol types were targeted, sliding window sizes were set to fit each symbol type, and there was little interference between objects contained in the tested P&IDs. Regarding the character recognition result, several character strings were recognized incorrectly as a one-character string (error case 3 of Figure 13b). In addition, an inclined character string was recognized as several independent character strings (error case 1 of Figure 13b) and a vertical character string showed low recognition accuracy compared to horizontal character strings (error case 4 of Figure 13b). The line recognition result showed difficulty in detecting dotted and diagonal lines (error case 1 of Figure 13c). The problem of recognizing dotted lines should be solved by determining the reference gaps between the line pieces forming a dotted line. This can be done by calculating the distribution of gaps between the horizontal line or the vertical line located on the same line. The problem of recognizing diagonal lines can be solved by determining the inclination

of the line at the outset, moving along the black pixels adjacent to the target black pixel, and grouping the adjacent black pixels together, which recognizes them as diagonal lines.

The symbol misidentification rate was 23.6% on average. The character misidentification rate was 19.2% on average. The line misidentification rate was 12.1%. The symbol recognition result revealed that symbols of a specific type were identified as an incorrect type (false positives) (error case 2 of Figure 13a). This problem can be solved by increasing the type and amount of training data and reinforcing the layers of the image-classification deep learning model accordingly. The character recognition result revealed that a symbol was recognized as a character (error case 2 of Figure 13b). This problem of multiple false positives can be solved by further training the CTPN using the character data extracted from the P&ID and modifying the CTPN model, considering the specificity of character strings included in P&ID. Regarding the line recognition result, several character strings were identified incorrectly as lines (error case 2 of Figure 13c).



(a) Symbol detection



(b) Character detection

Figure 13. *Cont.*

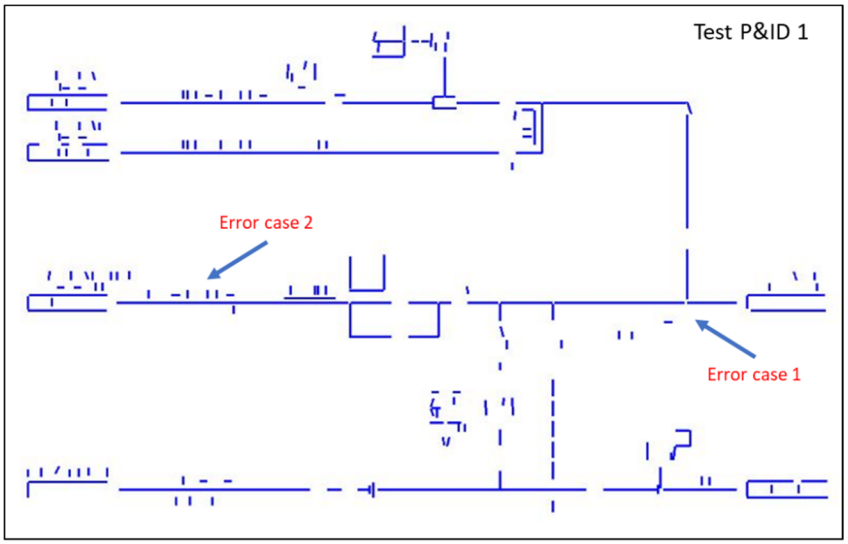


Figure 13. Recognition results of test P&IDs 1 and 2.

7. Conclusions

In this paper, we proposed a method to detect and extract various objects included in an image-format P&ID, which is crucial to converting it to a digital P&ID. The method consists of the preprocessing and recognition stages. In the preprocessing stage, diagram alignment, outer border removal, and title box removal are performed. In the recognition stage, symbols, characters, lines, and tables are detected. We used deep learning techniques in the process of symbol and character detection in a P&ID. Accordingly, we defined the training data structure required to develop a deep learning model for P&ID recognition. A new deep learning model for symbol detection is defined based on AlexNet. We also employed the connectionist text proposal network (CTPN) for character detection, and traditional image processing techniques for the P&ID line and table detection.

The main contributions of this study include: (a) the proposal of a general and comprehensive method for the recognition of the main objects used in P&ID drawings, (b) the development and application of P&ID object detection techniques tailored to each individual object type, and (c) the definition of the input data structure for efficient training of the deep learning model for P&ID recognition. We expect that this study contributes to the digitization of tons of paper-based P&IDs in the energy industry. In addition, this study can be employed in advanced maintenance technology using augmented reality, which recognizes objects from paper-based P&IDs [38].

The proposed method, however, has several limitations. To overcome these problems, we suggest the following measures.

- Limit the symbol types for detection of four types and nine classes. Considering that there are hundreds of symbol types used in P&IDs, there is a need to expand the symbol types intended for detection. In addition to the method that applies a sliding window to the image-classification deep learning model, it is necessary to examine the application of an image-detection deep learning model.
- With regard to character detection, expand the training data for a conventional text-detection deep learning model using text data extracted from P&IDs or improve the conventional deep learning models considering the particularity of the texts included in P&IDs.
- With regard to line detection, develop efficient methods to recognize dotted lines in addition to continued lines, and recognize diagonal lines in addition to horizontal and vertical lines. Furthermore, in the case where several lines intersect, a technique to determine their interconnectedness must be put in place. The proposed method is also prone to line recognition

errors when there are noises in the P&ID. Therefore, methods to apply a deep learning technique for line detection should be explored.

- For table detection, develop a method to recognize various table types and shapes, in addition to the most basic form used in this study.
- The training data used in the study was manually extracted from a real P&ID drawing. The absolute amount of training data was, therefore, far from being sufficient. This entailed constraints on the selection of applicable deep learning models. A follow-up study is needed to develop algorithms for automated detection and extraction of training data required for P&ID recognition.

For the analog to digital conversion of a P&ID, the following tasks must be tackled: (1) integration of characters associated with lines, (2) integration of characters associated with symbols, (3) identification of the line–symbol connection, (4) creation of structured P&IDs, (5) Tag ID and line number processing as per the item numbering system, (6) application of symbol catalogs, and (7) automated input of key attributes by symbol type using symbols, characters, lines, and tables detected in P&IDs.

Author Contributions: The contribution of the authors for this publication article are as follows: Conceptualization, D.M. Methodology, E.-S.Y. and J.-M.C. Software, E.-S.Y. Validation, T.L. Formal analysis, E.-S.Y. Writing—original draft preparation, E.-S.Y. Writing—review and editing, D.M. and J.K. Visualization, E.-S.Y. Supervision, D.M. Project administration, D.M. and funding acquisition, D.M.

Funding: The Industry Core Technology Development Program (Project ID: 20000725 & 10080662) funded by the Ministry of Trade, Industry and Energy and by the Basic Science Research Program (Project ID: NRF-2019R1F1A1053542) through the National Research Foundation of Korea (NRF) funded by the Ministry of Science and ICT of the Korean Government supported this study.

Acknowledgments: The authors would like to thank Kim, Sang Do a senior engineer and Cho, Won Hee a vice president at Samsung Engineering for providing image-format P&IDs.

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

References

1. Tornbre, K. Technical drawing recognition and understanding: From pixels to semantics. In Proceedings of the Workshop on Machine Vision and Application, Tokyo, Japan, 7–9 December 1992; pp. 393–401.
2. Fahn, C.-S.; Wang, J.-F.; Lee, J.-Y. A topology-based component extractor for understanding electronic circuit diagrams. *Comput. Vision Graph. Image Process.* **1988**, *43*, 279. [CrossRef]
3. Lee, S.-W.; Kim, J.H.; Groen, F.C. Translation-, Rotation- and Scale- Invariant Recognition of Hand-Drawn Symbols in Schematic Diagrams. *Int. J. Pattern Recognit. Artif. Intell.* **1990**, *4*, 1–25. [CrossRef]
4. Futatsumata, T.; Shichino, G.; Shibayama, J.; Maeda, A. Development of an Automatic Recognition System for Plant Diagrams. In Proceedings of the IAPR Workshop on Machine Vision Applications, Tokyo, Japan, 28–30 November 1990; pp. 207–210.
5. Benjamin, D.; Forgues, P.; Gulko, E.; Massicotte, J.; Meubus, C. The use of high-level knowledge for enhanced entry of engineering drawings. In Proceedings of the 9th International Conference on Pattern Recognition IEEE, Rome, Italy, 14 May 1988; pp. 119–124.
6. Kang, S.-O.; Lee, E.-B.; Baek, H.-K. A Digitization and Conversion Tool for Imaged Drawings to Intelligent Piping and Instrumentation Diagrams (P&ID). *Energies* **2019**, *12*, 2593.
7. Kato, H.; Inokuchi, S. The recognition method for roughly hand-drawn logical diagrams based on hybrid utilization of multi-layered knowledge. In Proceedings of the 10th International Conference on Pattern Recognition, Atlantic City, NJ, USA, 16–21 June 1990; pp. 578–582.
8. Maini, R.; Aggarwal, H. Study and comparison of various image edge detection techniques. *Int. J. Image Process.* **2008**, *3*, 1–12.
9. Xu, L.; Oja, E.; Kultanen, P. A new curve detection method: Randomized Hough transform (RHT). *Pattern Recognit. Lett.* **1990**, *11*, 331–338. [CrossRef]
10. Wang, D.; Haese-Coat, V.; Ronsin, J. Shape decomposition and representation using a recursive morphological operation. *Pattern Recognit.* **1995**, *28*, 1783–1792. [CrossRef]
11. Tian, D.P. A review on image feature extraction and representation techniques. *Int. J. Multimed. Ubiquitous Eng.* **2013**, *8*, 385–396.

12. Tan, W.C.; Chen, I.-M.; Tan, H.K. Automated identification of components in raster piping and instrumentation diagram with minimal pre-processing. In Proceedings of the IEEE International Conference on Automation Science and Engineering (CASE), Fort Worth, TX, USA, 21–25 August 2016; pp. 1301–1306.
13. Arroyo, E.; Hoernicke, M.; Rodriguez, P.; Fay, A. Automatic derivation of qualitative plant simulation models from legacy piping and instrumentation diagrams. *Comput. Chem. Eng.* **2016**, *92*, 112–132. [CrossRef]
14. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet Classification with Deep Convolutional Neural Networks. In Proceedings of the NIPS 2012: Neural Information Processing Systems Conference, Lake Tahoe, NV, USA, 3–6 December 2012.
15. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 7–12 June 2015; pp. 1–9.
16. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 779–788.
17. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J.; Malik, J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, USA, 23–28 June 2014; pp. 580–587.
18. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot Multibox Detector. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 21–37.
19. Sliding Windows for Object Detection with Python and openCV. Available online: https://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/ (accessed on 24 September 2019).
20. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440.
21. TensorFlow. Available online: https://www.tensorflow.org/ (accessed on 24 September 2019).
22. Keras. Available online: https://keras.io/ (accessed on 24 September 2019).
23. PyTorch. Available online: https://pytorch.org/ (accessed on 24 September 2019).
24. Theano. Available online: http://deeplearning.net/software/theano/# (accessed on 24 September 2019).
25. LeCun, Y.; Boser, B.; Denker, J.S.; Henderson, D.; Howard, R.E.; Hubbard, W.; Jackel, L.D. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Comput.* **1989**, *1*, 541–551. [CrossRef]
26. Fu, L.; Kara, L.B. From engineering diagrams to engineering models: Visual recognition and applications. *Comput. Des.* **2011**, *43*, 278–292. [CrossRef]
27. Rahul, R.; Paliwal, S.; Sharma, M.; Vig, L. Automatic Information Extraction from Piping and Instrumentation Diagrams. In Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods, Prague, Czech Republic, 19–21 February 2019; pp. 163–172.
28. *ISO 10628: Flow Diagrams for Process Plants—General Rules*; International Organization for Standardization: Geneva, Switzerland, 1997.
29. Everingham, M.; Van Gool, L.; Williams, C.K.; Winn, J.; Zisserman, A. The pascal visual object classes (voc) challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
30. Yu, E.S. Approach of Object Recognition from Image Format Engineering Drawings using Deep Learning. Master's Thesis, Kyungpook National University, Daegu, Korea, 2019.
31. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556.
32. 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 (CVPR), Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
33. LeCun, Y.; Bottou, L.; Bengio, Y.; Haffner, P. Gradient-based learning applied to document recognition. *Proc. IEEE* **1998**, *86*, 2278–2324. [CrossRef]
34. Krizhevsky, A.; Hinton, G. *Learning Multiple Layers of Features from Tiny Images*; Technical Report; University of Toronto: Toronto, ON, Canada, 2009; p. 7.
35. Tian, Z.; Huang, W.; He, T.; He, P.; Qiao, Y. Detecting Text in Natural Image with Connectionist Text Proposal Network. In *European Conference on Computer Vision*; Springer: Cham, Switzerland, 2016; pp. 56–72.

36. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
37. Karatzas, D.; Shafait, F.; Uchida, S.; Iwamura, M.; i Bigorda, L.G.; Mestre, S.R. ICDAR 2013 Robust Reading Competition. In Proceedings of the 12th International Conference on Document Analysis and Recognition, Washington, DC, USA, 25–28 August 2013; pp. 1484–1493.
38. Gattullo, M.; Evangelista, A.; Uva, A.E.; Fiorentino, M.; Boccaccio, A.; Manghisi, V.M. Exploiting Augmented Reality to Enhance Piping and Instrumentation Diagrams for Information Retrieval Tasks in Industry 4.0 Maintenance. In *International Conference on Virtual Reality and Augmented Reality*; Springer Science and Business Media LLC: Cham, Switzerland, 2019; pp. 170–180.