Deep learning for digitising complex engineering drawings.

JAMIESON, L.

2024

Deep Learning for Digitising Complex Engineering Drawings

Laura Jamieson

A report submitted as part of the requirements for the degree
of PhD in Computing
at the School of Computing
Robert Gordon University
Aberdeen, Scotland

October 2024

Supervisor Professor Eyad Elyan

Abstract

Vast amounts of documents are still commonly stored in undigitised formats. Consequently, the data they contain cannot be used to its full potential, as substantial manual effort is required to analyse it. Amongst these documents, engineering drawings are considered one of the most challenging to digitise. The task involves automatically recognising all drawing components which are the symbols, text and connections. Although there has been significant improvement in computer vision due to the development of deep learning, the same progress has not been seen for engineering drawing digitisation. Most of these methods were based on traditional approaches which require manual feature selection and heuristics.

This thesis presents a deep learning framework for the challenging problem of digitising complex engineering drawings. This is a fully automated approach for the processing and analysis of these drawings. It contains a set of deep learning methods for digitising the different drawing components.

New methods were presented to recognise engineering symbols. Text digitisation methods were also developed. It should be noted that this represents a substantially more challenging problem compared to text digitisation in typical documents, due to reasons such as the varying text locations, orientations, and text strings often being composed of codes instead of known words.

The thesis has solved inherent challenges in the field of engineering drawing digitisation. Furthermore, the thesis has opened up a new direction towards addressing the data annotation problem, by using few-shot learning for symbol detection.

All of the methods presented here have been thoroughly tested on real world complex engineering drawings from different domains. These were Piping and Instrumentation diagrams from the oil and gas industry, and multiple engineering drawing types from the construction industry.

Keywords: Engineering Drawing, Digitisation, Deep Learning, Symbol Detection,

Text Detection, Text Recognition, Few-Shot Learning, Multiclass Imbalanced Classification

Acknowledgements

Thanks go to Professor Eyad Elyan and Dr Carlos Moreno-Garcia for the supervision. I would also like to thank family and friends for their support.

The author would like to thank DNV GL and TaksoAi for their support during this study by providing domain expertise knowledge and data for validating the work.

Abbreviations

AAI	Applied Artificial Intelligence
ACV	Automatic Control Valve
AI	Artificial Intelligence
AP	Average Precision
AUC	Area Under the Curve
bAP	base Average Precision
BIM	Building Information Modelling
BOM	Bill Of Materials
CC	Connected Components
CCA	Connected Component Analysis
CLEval	Character Level Evaluation
CNN	Convolutional Neural Network
COCO	Common Objects in Context
CRAFT	Character Region Awareness For Text detection
CRNN	Convolutional Recurrent Neural Network
CTC	Connectionist Temporal Classification
CTPN	Connectionist Text Proposal Network
CVAT	Computer Vision Annotation Tool
DAC	Deep Adaptive image Clustering
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DGCNN	Dynamic Graph Convolutional Neural Network
dpi	dots per inch
EAST	Efficient and Accurate Scene Text detector
ED	Engineering Drawing
Fast R-CNN	Fast Region-based CNN
Faster R-CNN	Faster Region-based CNN
FCN	Fully Convolutional Network
FN	False Negative
FP	False Positive

FPN	Feature Pyramid Network
fps	frames per second
FSC	Few-Shot Classification
FSL	Few-Shot Learning
FSOD	Few-Shot Object Detection
FS-Symbol	Few-Shot-Symbol
GAN	Generative Adversarial Network
GCN	Graph Convolutional Network
GNN	Graph Neural Network
HOG	Histogram of Oriented Gradient
HVAC	Heating, Ventilation and Air Conditioning
ICDAR	International Conference on Document Analysis and Recognition
IJCNN	International Joint Conference on Neural Networks
IJDAR	International Journal on Document Analysis and Recognition
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IOU	Intersection Over Union
LBP	Local Binary Pattern
LMI	Locally Mounted Instrument
LSTM	Long Short-Term Memory
mAP	mean Average Precision
MFC-GAN	Multiple Fake Class GAN
MSER	Maximally Stable Extremal Regions
nAP	novel Average Precision
NMS	Non Maximum Suppression
OCR	Optical Character Recognition
P&ID	Piping and Instrumentation Diagram
RAG	Region Adjacency Graph
R-CNN	Region-based CNN
RF	Random Forest
R-FCN	Region-based FCN
RNN	Recurrent Neural Network
ROI	Regions of Interest
RPN	Region Proposal Network
SESYD	Systems Evaluation SYnthetic Documents
SiED	Symbols in Engineering Drawings dataset
SIFT	Scale Invariant Feature Transform
SMOTE	Synthetic Minority Oversampling Technique
SSD	Single Shot Detector

SURF	Speeded Up Robust Features
SVM	Support Vector Machine
SWT	Stroke Width Transform
TBMSL-Net	Three Branch and Multi-Scale Learning Network
TFA	Two-stage Fine-tuning Approach
TGS	Text/Graphics Separation
TP	True Positive
V&C Provision	Valve and Capped Provision
VOC	Visual Object Classes
YOLO	You Only Look Once

Publications

Journals

- Jamieson L, Moreno-Garcia C.F, Elyan E. A Review of Deep Learning Methods for Digitisation of Complex Documents and Engineering Diagrams. In Artificial Intelligence Review journal, 57, 2024: 1-37, https://doi.org/10.1007/s10462-024-10779-2
- Jamieson L, Moreno-Garcia C.F, Elyan E. Towards Fully Automated Processing and Analysis of Construction Diagrams: AI-Powered Symbol Detection. In International Journal on Document Analysis and Recognition (IJDAR), 2024, https://doi.org/10.1007/s10032-024-00492-9
- Jamieson L, Elyan E, Moreno-Garcia C.F, Few-Shot Symbol Detection in Engineering Drawings. In Applied Artificial Intelligence (AAI) journal, 38(1), 2024, https://doi.org/10.1080/08839514.2024.2406712.

Conferences

- Jamieson L, Moreno-Garcia C.F, Elyan E. Deep Learning for Text Detection and Recognition in Complex Engineering Diagrams. In 2020 International Joint Conference on Neural Networks (IJCNN). IEEE. https://doi.org/10.1109/IJCNN48605.2020.9207127
- Jamieson L, Moreno-Garcia C.F, Elyan E. A Multiclass Imbalanced Dataset Classification of Symbols from Piping and Instrumentation Diagrams. In 2024 International Conference on Document Analysis and Recognition (ICDAR). https://doi.org/10.1007/978-3-031-70533-5_1

Indirect Contributions

- Elyan E, Jamieson L, Ali-Gombe A. Deep learning for symbols detection and classification in engineering drawings. In Neural Networks journal, 129, 2020: 91-102, https://doi.org/10.1016/j.neunet.2020.05.025

Declaration

I confirm that the work contained in this PhD project report has been composed solely by myself and has not been accepted in any previous application for a degree. All sources of information have been specifically acknowledged and all verbatim extracts are distinguished by quotation marks.

Signed ..... Date .....

Laura Jamieson

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter provides an introduction to the problem of Engineering Drawing (ED) digitisation. This starts with a discussion of the purpose of these drawings, which also highlights their wide use throughout numerous industry sectors. Then, it goes on to describe why there is a significant demand to digitise these drawings. Background information on previous research in the area is provided. Next, the motivations behind this research are thoroughly discussed. This is followed by detailed research objectives and descriptions of the thesis contributions. An outline of the thesis structure is also provided in this chapter.

1.1 Engineering Drawings

EDs are widely used in a range of industries including oil and gas [1], architecture [2] and nuclear [3]. They represent equipment, its connections and text annotations. Consequently, a vast amount of information is often stored in these drawings. Their analysis is frequently required for a wide range of purposes, such as crucial safety studies [3]. However, as they are commonly stored in an undigitised file format or as a paper copy, the data contained in these drawings is not readily accessible to use, and analysing it requires substantial manual effort. Furthermore, often a whole dataset of drawings will need to be inspected. However, manual analysis is very time-consuming [4], requires subject matter experts [5], and is open to human error [6] and individual interpretations [7].

Digitising EDs is a problem that has attracted the attention of researchers for several decades [8, 9, 10, 11, 12]. The task involves automatically recognising the shapes in an undigitised drawing using computer vision methods. The output is then collated to create a digitised list of recognised items in the drawing. Initial digitisation approaches

were based on traditional methods [13]. These required manual feature engineering to transform raw data into a suitable representation from which patterns can be found [14]. However, they fail to generalise well to unseen scenarios and thus were not suitable for application to the large range of appearance variations seen in EDs, such as orientation, scale and image degradation. For traditional methods to perform well for the drawing digitisation task, manual fine tuning and carefully designed rules would be required for every appearance variation. Moreover, all potential scenarios must be known in advance, which is typically unfeasible. For instance, in one study using traditional methods, electrical symbols were recognised from scanned logic circuit diagrams using morphological operations, however the method was not able to detect the broken symbols [15]. In another example, the Hough transform was used to extract data from EDs, however it did not perform well due to presence of noise, overlapping characters and manual annotations on the drawing [16].

Deep learning methods have made a significant improvement to the field of computer vision [14]. This was influenced by several factors, including the relatively recent increases in computer power and amounts of available data. It was also largely driven by the development of Convolutional Neural Networks (CNN), which were shown to automatically learn relevant features from image data and outperform traditional approaches [17, 18]. These models have advanced various computer vision techniques including those of object detection, text detection and text recognition. For example, various object detection models, such as the You Only Look Once (YOLO) models [19, 20, 21, 22, 23, 24, 25, 26], have been proposed within the last decade. These models are designed to learn features automatically from data during training. Another benefit compared to traditional methods is that they generalise better to unseen data.

However, the significant progress in computer vision has not been reflected in the field of ED digitisation. Deep learning methods were only used for this in the last five years. Furthermore, the problem presents specific research challenges [27]. For example, much research in object detection focusses on the detection of objects in relatively small coloured images [28]. In contrast, EDs are considerably larger grayscale images, with objects that can be represented by only a few shapes and thus have limited features to learn from.

There are many challenges associated with digitising these documents. This includes the acquisition of ED datasets, which are not readily available in the public domain, largely due to the confidential information they contain [29, 30]. Additionally, most deep learning models require a large labelled dataset to learn from. The annotation task for EDs is not trivial and in many cases, this makes it unviable to implement deep learning models. Another challenge arises from the fact that deep learning models

are typically designed to learn from balanced data, whereas ED datasets are usually imbalanced. For instance, symbol datasets are inherently imbalanced due to the uneven representation of equipment within the drawings [31, 32]. In these cases, the class imbalance problem arises, which is when deep learning models are biased towards the majority classes [33].

1.2 Motivation

Across a range of industries, it is still common to store documents in an undigitised format. Consequently, the data contained in these documents cannot be used to its full potential, as substantial manual effort is required to search and access it. Amongst these documents, EDs are one of the most challenging to digitise due to their complexity [1]. These drawings typically contain numerous component types and overlapping elements which are often shown in dense representation. Currently, this data is underutilised as it is locked away in undigitised files and cannot be readily used in further analysis. Therefore, developing improved methods for ED digitisation would improve access to a substantial amount of information.

Whilst deep learning has significantly improved the field of computer vision [14], the progress in ED digitisation has been relatively slow [27]. Most digitisation methods in this field were based on traditional image processing approaches [13]. These methods need to be individually designed for each different scenario, as they use carefully selected heuristics which do not generalise well outwith controlled use cases. Furthermore, much of the previous research in this area has utilised small datasets of simplified drawings relative to those found in industry [34, 8, 4].

This research will evaluate digitisation methods using large datasets of real-world drawings sourced from industry. These datasets contain drawings of various formats and qualities. In addition, these will be of different drawing types, specifically, Piping and Instrumentation Diagrams (P&ID) and construction diagrams, which are from the oil and gas and construction sectors respectively. These are very complex drawings and contain various shapes which often overlap. An example of part of a P&ID is shown in Figure 1.1. It is important to use industry sourced drawings for this research, as unlike simplified versions, they accurately reflect the challenges associated with real-world ED digitisation. Furthermore, using large datasets of varied drawings will enable methods to be evaluated across a wide range of use cases.

In this thesis, there was a specific focus on the digitisation of two of the most crucial drawing elements, which are the text and the symbols. Text is used throughout the diagrams to convey critical information such as operating details. Text digitisation

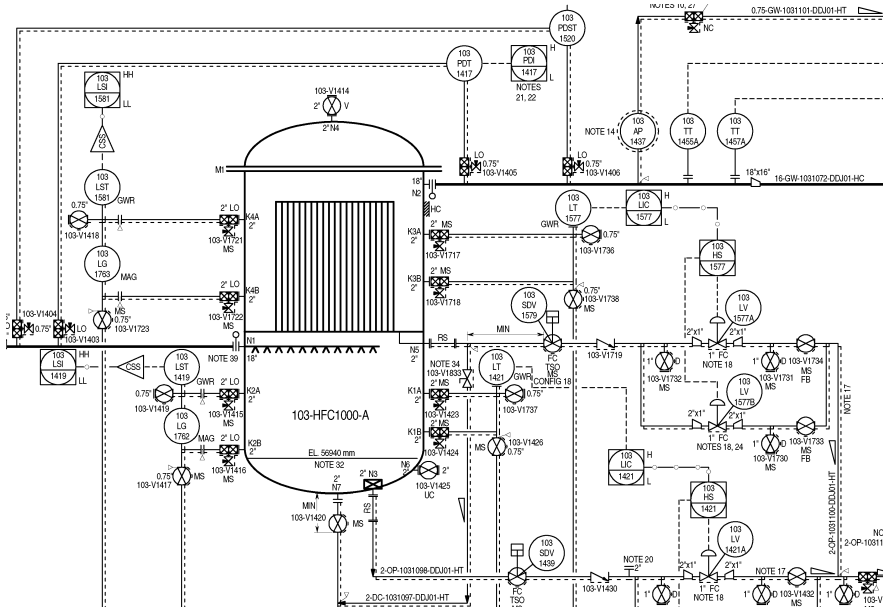


Figure 1.1: Part of a P&ID

represents a challenging problem as the text can be located anywhere in the drawing, and of multiple different fonts and orientations. The symbol data is important as these shapes represent various engineering equipment within the drawings. Symbols are challenging to digitise for multiple reasons [11], for example there can be on average 180 symbols per diagram [1] from numerous classes. Furthermore, they can be represented by only a few lines, have varying orientation, and often contain shapes similar to that in the rest of the drawing. This research therefore includes the topics of text detection, text recognition and symbol detection. Additionally, it involves the research challenges of learning from only a few instances, and multiclass imbalance classification.

The developed frameworks were deployed in several companies. To investigate human-in-the-loop methods [35] in this real-world scenario, the frameworks were designed to be interactive. This allows for the method predictions to be manually reviewed and for incorporation of human knowledge into the output. For example, the user is able to manually add an undetected symbol, or remove a false positive symbol from the output file. Furthermore, the framework also allows the manually reviewed data to be used in an iterative training process.

To summarise, although deep learning has brought significant change in the field of computer vision, the same cannot yet be said for ED digitisation methods. This problem is still considered very challenging [13]. There is significant demand from both industry and the research community to develop improved digitisation methods and improve access to a vast amount of data [1, 2, 29]. There is large potential to use deep learning

methods to improve digitisation methods in this field. However, many challenges exist in this domain and further research is needed to solve this challenging problem. Due to the wide use of these drawings, this work has potential applications across many sectors.

1.3 Research Objectives

The objectives of this research are as follows:

- To identify the main challenges and research gaps in the field of deep learning for engineering drawing digitisation. This will involve critically reviewing the existing literature to identify the methods that have been presented for the digitisation of all the components in these drawings.
- To create a novel experimental framework for the processing and analysis of EDs. This framework is a set of methods for the automatic recognition of symbols and text within these drawings. Extensive experiments on real world datasets of different ED types of various qualities will be carried out.
- To address one of the most challenging tasks in ED digitisation, that of data annotation, using few-shot learning. This approach significantly reduces the need for a large labelled training dataset of engineering symbols as it requires only a few labelled training instances per novel class. Extensive experiments on a P&ID dataset will be completed to evaluate the method's applicability. Furthermore, the method will be compared to state-of-the-art object detection approaches.
- To address the challenge of lack of real-world data by presenting a symbol dataset. This will contain a large number of symbols from a variety of classes and it will be imbalanced which reflects the symbol distribution seen in the real world. Additionally, a method to improve multiclass imbalanced classification will be presented.

1.4 Contributions

The main contributions of this thesis are outlined as follows:

- A critical and comprehensive investigation of the deep learning-based methods for complex ED digitisation. This includes a thorough discussion of the open research challenges associated with deep learning solutions for these drawings, which were identified as dataset availability, data annotation, evaluation, class imbalance and contextualisation. Recommendations for future research directions

are also presented to overcome these challenges. *This work has been published in the Artificial Intelligence Review journal [36].*

- A thorough discussion and evaluation of deep learning methods for text digitisation in complex real world P&IDs sourced from the oil and gas industry. This includes an evaluation of a deep learning text detection method, specifically the Efficient and Accurate Scene Text detector (EAST) for its applicability to EDs. Additionally, a Long Short-Term Memory (LSTM) network based method was evaluated for its performance on text recognition. *This work was presented at the 2020 IJCNN [37].*

- A novel framework for the automatic processing and analysis of EDs. This detects symbols for the task of material takeoff in construction diagrams, resulting in significant time-saving compared to manual drawing analysis. Extensive experiments were carried out using a large dataset of challenging high-resolution drawings of different qualities. Various symbol classes were used, which had high levels of intra-class variability and inter-class similarity. This is believed to be the first example of these experiments using complex construction diagrams from industry. The methods were based on two state-of-the-art object detection architecture types, one-stage and two-stage. *This work has been published in the IJDAR [38].*

- A few-shot symbol detection approach for EDs. This is one of the first examples of few-shot methods used for real-world complex drawings. This method is particularly beneficial for rare symbols and allows for additional classes to be incorporated into a symbol detection model with only a few labelled samples. Fewer than ten samples, and as low as one sample, per novel class were used. Extensive experiments were carried out to validate the approach, and the results indicate that the method shows statistically significant improvement compared to other state-of-the-art detection methods. *This work has been published in the AAI journal [39].*

- A new multiclass imbalanced dataset of symbols from real-world EDs, specifically P&IDs, is presented [1]. This dataset contains $7,728$ symbols from 48 classes and it is considered one of the first of its kind in the research community. Furthermore, a method for handling multiclass imbalance classification of these symbols is provided. This is based on class decomposition by means of unsupervised machine learning methods. Experiments using CNNs showed that using class decomposition significantly improves the classification performance that can be achieved,

[1] https://github.com/carlosfmorenog/CDSMOTE-NONBIN-Symbols

without causing information loss, as it is the case with other class imbalance data sampling approaches. *This work was presented at the 2024 ICDAR [40].*

A list of publications that resulted from these contributions is presented in the preface section Publications.

1.5 Thesis Structure

The remainder of this thesis is structured as follows. Chapter 2 is the Literature Review. This section critically reviews the existing deep learning methods used for ED digitisation, including methods for symbol recognition, text extraction and connection detection. Chapter 3 is Text Digitisation. Here, a thorough evaluation of deep learning methods applied for text digitisation in complex P&IDs is presented. Extensive experiments were performed to demonstrate where the methods perform well and which scenarios present challenges. This is followed by Chapter 4 which is Symbol Detection. This chapter introduces a novel framework for the automatic processing of construction drawings. Next is Chapter 5, which is Few-Shot Symbol Detection. This presents a few-shot framework for symbol detection in real-world EDs. Thorough experiments are presented using fewer than ten labelled samples per novel class. This is followed by Chapter 6, which is Multiclass Imbalanced Symbols Classification. A multiclass dataset of engineering symbols is provided and made available in the public domain. Additionally, a method for multiclass imbalanced symbols classification based on class-decomposition is presented. Finally, Chapter 7 presents the Conclusion and Future Work. The research and findings are also summarised, followed by a discussion of the limitations and suggestions for future research directions.

Chapter 2

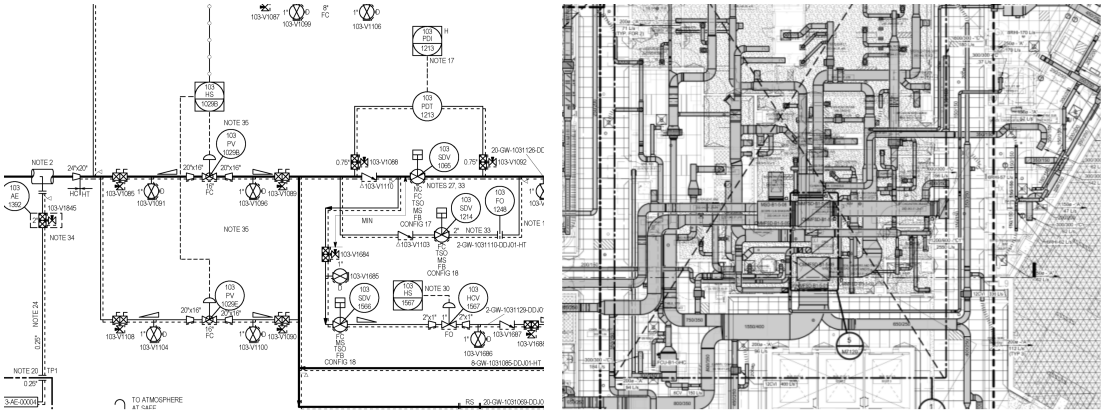
Literature Review

This chapter provides a comprehensive and critical review of the existing literature that presents deep learning based methods for ED digitisation. This is presented in the context of a wide range of applications across different industry sectors. This includes an in-depth technical discussion of state-of-the-art methods for handling symbols, text, and connectivity information in these diagrams. Remaining challenges in this area are then identified as dataset availability, data annotation, evaluation methods, class imbalance and contextualisation. These challenges are all outlined and thoroughly discussed here. *This work has been published in the Artificial Intelligence Review journal [36].*

2.1 Introduction

EDs are considered one of the most complex document types to digitise. This is due to multiple reasons such as the combination of vast variety of symbols and text, dense representation of equipment and non standard formatting. Furthermore, there can be scientific annotations and the drawings can be edited over time to contain annotations from multiple disciplines. These diagrams are prevalent across multiple industries, including electrical [41], oil and gas [1], and architecture [42]. Manual analysis of these diagrams is time-consuming, prone to human error [43, 6] and requires subject matter experts [43]. There has recently been an increasing demand to digitise these diagrams for use in processes including asset performance management [44], safety studies [3], and data analytics [13]. Due to its importance, the problem of complex diagram digitisation is receiving interest from academia and industry [27, 45]. For instance, engineering was the field with the most recent digitalisation-related publications in the Scopus database [46]. EDs are complex and used for different purposes, as seen in Figure 2.1. Figure 2.1a represents part of a P&ID. These are commonly used in offshore oil and gas installations,

while Figure 2.1b presents part of a Heating, Ventilation and Air Conditioning (HVAC) diagram, commonly utilised in construction projects.



a) b)

Figure 2.1: a) Small section of a P&ID b) Small section of a HVAC diagram

Various methods have been developed over the past four decades to automate the processing, analysing and interpretation of these diagrams [8, 9, 10, 11, 12]. A relatively recent review by Moreno-Garcia et al. [13] showed that most relevant literature followed a traditional machine learning approach to automate these drawings. Traditional approaches are based on hand-crafting a set of features which are then input to a specific supervised machine learning algorithm [17]. Extensive feature engineering and expert knowledge were often required to design suitable feature extractors [17]. Image features were typically based on colour, edge and texture. Examples of commonly used image features include Histogram of Oriented Gradient (HOG) [47], Scale Invariant Feature Transform (SIFT) [48], Speeded Up Robust Features (SURF) [49] and Local Binary Pattern (LBP) [50]. The feature vectors were classified using algorithms, such as a Support Vector Machine (SVM). Whilst traditional methods were shown to work well in specific use cases, they were not suited to the extensive range of characteristics present in EDs [13]. For example, traditional symbol classification methods may be limited by variations in symbol appearance, including rotation, translation and degradation [13]. Morphological changes and noise also compromised traditional methods' accuracy [51]. The reliance of traditional methods on pre-established rules resulted in weak generalisation ability across variations [52].

In recent years, deep learning has significantly advanced the domain of computer vision [14]. Deep learning is a subfield of machine learning, which is itself a subfield of artificial intelligence. Figure 2.2 illustrates the key differences between traditional and deep learning methods. In contrast to traditional machine learning-based methods, deep learning-based methods learn features automatically. Deep learning models contain

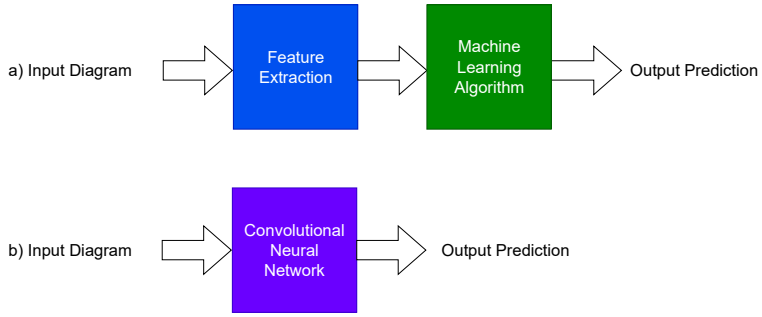


Figure 2.2: Comparison of traditional and deep learning approaches for ED digitisation a) traditional approach b) deep learning approach

multiple computation layers which can be trained to extract relevant features from data. CNN have improved computer vision methods, including image classification, segmentation and object detection [14]. In 1998, LeCun et al. [17] introduced the influential LeNet model. The authors presented a CNN-based method for handwritten character recognition. They showed that a CNN could automatically learn features from pixel data and outperform traditional approaches. However, a significant improvement in methods was seen mainly since 2012 when Krizhevsky et al. [18] presented the AlexNet model. AlexNet was used to classify images in the 2012 ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [28]. The authors obtained the winning score by a large margin. The top 5 error rate was 15.3%, compared to 26.2% for the second-place method. Since then, there has been a considerable rise in deep learning. This was facilitated by algorithm developments, improvements in computing hardware, and a significant increase in available data.

Despite the recent and unprecedented progress, digitising EDs continues to be a challenging problem [13]. First of all, these diagrams are very complex, containing a large number of similar [43, 34] and overlapping [34] shapes. For example, Elyan et al. [1] reported on average 180 symbols of different types in a real-world P&ID dataset. The presence of text is another challenging problem. There is no consistent pattern for engineering equipment layout, meaning the text can be present anywhere in the diagram. It is also commonly present in multiple fonts [34], scales and orientations [3]. Contextualisation of the extracted data is a further challenge. This involves determining the relationships between extracted data, for example, associating a tag with the relevant symbol. Moreno-Garcia and Elyan [27] identified three additional challenges as document quality, imbalanced data and topology. Although a large proportion of the related literature analysed high-quality drawings, in practice, the drawings can be

low-quality [27]. Another factor restricting the development of deep learning models in this area is the lack of publicly available datasets [45, 13]. Furthermore, annotation of these datasets is required for use with supervised learning algorithms, which is typically a time-consuming and often impractical manual process.

Here, we present a comprehensive critical investigation of existing literature that utilises state-of-the-art deep learning methods for digitising complex EDs. In a related area, Pizarro et al. [53] provided a review on the automatic analysis and recognition of floor plans. They focussed on both rule-based and learning-based approaches. However, there is a gap in the literature, as there is no published review which covers the surge in the deep learning research in ED digitisation published in the last five years.

The reviewed literature was selected according to several criteria. First, the paper should present a deep learning method for the digitisation of EDs. This covers a wide variety of drawing types, such as P&IDs and architectural diagrams. This review also covers the literature that focussed on the digitisation of specific elements, such as presenting a detection method for symbols, as well as that which presented multiple methods to digitise more than one diagram component. Papers which presented a mixture of deep learning and traditional methods were included. Second, we reviewed peer-reviewed articles from academic databases including IEEE Xplore, ACM Digital Library and Science Direct. Third, we focus on the recent literature that was published in the last five years. This shows there is an urgent need for more accurate and stable methods to handle such complex documents and EDs. Furthermore, from analysing these papers, remaining challenges were elicited, which were datasets, data annotation, evaluation, class imbalance and contextualisation.

2.2 Related Work

Deep learning has been used for diagram digitisation across various domains. The diagrams are composed of three elements. These are symbols, text and connectors. Connectors link symbols together and represent various line types, including continuous or dashed lines. Specialised computer vision methods are required to digitise each element type. This section introduces and discusses the application domains, together with the state-of-the-art deep learning methods used in the recent and relevant literature on complex ED digitisation.

2.2.1 Application Domains

The reviewed literature is listed by application and extracted data type in Table 2.1. Amongst these applications, there has been a considerable research focus on P&IDs

[34, 54, 51, 44, 3, 1, 55, 37, 11, 43, 56, 32, 57, 6, 58, 59, 45]. Another research area is architecture diagram digitisation [60, 52, 61, 42, 62, 2]. Deep learning methods were also applied to technical drawings [63], construction drawings [64] engineering documents [65] and diagrams.

Table 2.1: Relevant literature by application domain and extracted data

Reference	Year	Application	Extracted Data		
			Symbols	Text	Connectors
Ziran and Marinai [60]	2018	architectural	✓	-	-
Rahul et al. [34]	2019	P&IDs	✓	✓	✓
Sinha et al. [54]	2019	P&IDs	-	✓	-
Yu et al. [51]	2019	P&IDs	✓	✓	✓
Renton et al. [66]	2019	floor plans	✓	-	-
Mani et al. [44]	2020	P&IDs	✓	✓	-
Gao et al. [3]	2020	P&IDs	✓	✓	-
Elyan et al. [1]	2020	P&IDs	✓	-	-
Zhao et al. [52]	2020	architectural	✓	-	-
Rezvanifar et al. [61]	2020	architectural	✓	-	-
Moreno-Garcia et al. [55]	2020	P&IDs	✓	✓	✓
Jamieson et al. [37]	2020	P&IDs	-	✓	-
Nurminen et al. [11]	2020	P&IDs	✓	-	-
Paliwal et al. [43]	2021	P&IDs	✓	✓	✓
Moon et al. [56]	2021	P&IDs	-	-	✓
Nguyen et al. [63]	2021	technical drawings	✓	✓	-
Kim et al. [32]	2021	P&IDs	✓	✓	-
Stinner et al. [57]	2021	P&IDs	✓	-	✓
Paliwal et al. [6]	2021	P&IDs	✓	-	-
Hu et al. [67]	2021	mechanical drawings	✓	✓	-
Joy and Mounsef [4]	2021	electrical engineering	✓	✓	-
Schiebel et al. [68]	2021	EDs	-	✓	-
Kim et al. [42]	2021	architectural	✓	✓	-
Renton et al. [62]	2021	architectural	✓	-	-
Toral et al. [58]	2021	P&IDs	✓	✓	-
Mizanur Rahman et al. [69]	2021	circuit diagrams	✓	-	-
Hantach et al. [45]	2021	P&IDs	✓	✓	✓
Bickel et al. [70]	2021	principle sketches	✓	-	-
Bhanbhro et al. [59]	2022	P&IDs	✓	-	-
Sarkar et al. [71]	2022	EDs	✓	✓	-
Francois et al. [65]	2022	engineering documents	-	✓	-
Jakubik et al. [2]	2022	architectural	✓	✓	-
Gupta et al. [72]	2022	P&IDs	✓	-	-
Bickel et al. [73]	2023	principle sketches	✓	-	-
Mafipour et al. [74]	2023	technical drawings	✓	✓	-
Haar et al. [75]	2023	engineering and manufacturing drawings	✓	✓	-
Rumalshan et al. [76]	2023	railway technical maps	✓	✓	-
Theisen et al. [29]	2023	process flow diagrams	✓	-	✓

Most of the P&ID digitisation literature focussed on the extraction of specific data types [54, 3, 1, 37, 11, 56, 32, 57, 6, 58]. There is a particular focus on P&ID symbols [1, 11, 6]. For example, Elyan et al. [1] presented a YOLOv3 [21] based detection method for symbols in real-world P&IDs. A Generative Adversarial Network (GAN) based [77] approach was used to synthesise more data to improve classification. Meanwhile, Paliwal et al. [6] used a graph-based approach for symbol recognition. Other studies focussed on the text [37, 65] or connectors [56]. Studies that presented methods for

multiple element types were also seen [3, 57]. For instance, Gao et al. [3] created a Region-based FCN (R-FCN) [78] component detection method and a SegLink [79] based text detection method. Meanwhile, Stinner et al. [57] presented work on extracting symbols, lines and line crossings, however they did not consider the text.

There are only a few recent P&ID digitisation studies that presented methods for symbols, text and connectors [43, 34, 51, 44, 45]. These were often focused on specific elements of interest. For example, Mani et al. [44] created symbols, text and connection detection methods. They considered two symbol classes and recognised the text associated with these symbols. Hantach et al. [45] also proposed symbol, text and lines methods. The authors only had access to a limited dataset of eight P&IDs and considered one symbol class. Meanwhile, Yu et al. [51] created methods for tables aswell as symbols, lines and text. Deep learning was used for symbols and text, while the lines and table detection methods were based on traditional image processing.

Extracted elements have been associated to each other using distance-based or graph-based methods [44, 43, 34, 73, 29]. For instance, Mani et al. [44] determined symbol-to-symbol connections by representing the P&ID in graph format and implementing a depth-first search. Paliwal et al. [43] used a graph-based method to associate lines with relevant symbols and text. Meanwhile, Rahul et al. [34] used the Euclidean distance to associate detected symbols, tags and pipeline codes with the closest pipeline. Theisen et al. [29] presented methods for the digitisation of process flow diagrams. They used a Faster Region-based CNN (Faster R-CNN) [80] model to detect the unit operations, and a pixel search based algorithm to detect the connections between them. Then, the data was converted to a graph.

Deep learning has also been recently applied for the digitisation of architecture diagrams [60, 52, 61, 42, 62, 2]. These present similar challenges to engineering diagrams, such as various semantically equivalent symbol representations [61], relatively small objects [42] and the presence of occlusion and clutter [61]. One example is the work by Zhao et al. [52], which proposed a YOLO [19] based method to detect components in scanned structural diagrams. The authors suggested the method as a basis for Building Information Modelling (BIM). Various approaches have been presented for symbol detection in floor plans, including YOLO [61], Faster R-CNN [2, 60] and graph-based [62] methods.

There are a wide variety of uses of the digitised diagram data. This includes similarity search [73], diagram comparison [81] and classification [82]. For instance, Van Daele et al. [81] used deep learning to create a technical diagram similarity search tool [81]. They used $5,000$ technical diagrams. A traditional method based on Density-Based

Spatial Clustering of Applications with Noise (DBSCAN) [83] was used to partition the diagram. A CNN containing three convolutional layers classified drawing segments as 'table', 'two-dimensional CAD drawing' or 'irrelevant'. A siamese neural network classified a pair of CAD images as either 'same' or 'different' based on cosine similarity. An accuracy of 96.9% was reported.

Xie et al. [82] used deep learning to classify engineering diagrams according to the manufacturing method. A dataset of 1692 industry diagrams of engineering equipment was used. First, the diagrams were pre-processed by removing tables and dimension lines. Information tables were identified using CascadeTabNet [84]. The model contained two neural networks. The first, HRNet, was used for feature extraction and the second, Cascade R-CNN, for bounding box proposal. Reported precision was 97%. In comparison, the precision of a heuristic method based on watershed segmentation was lower at 78%. Dimension lines were detected using a Graph Neural Network (GNN), which outperformed a heuristic method. However, the authors reported that the network predictions allowed higher fault tolerance. The pre-processed diagram was then converted to graph format. Each node was embedded with line start and end positions. A GNN was used to predict the appropriate manufacturing method. This was shown to outperform various CNN and graph-based approaches. Overall accuracy of 90.8% was reported.

Digitised data from engineering diagrams can be used towards creating a digital twin [85], [74]. For instance, Vilgertshofer et al. [85] created a CNN-based symbol detection method to check for discrepancies between archived railway technical drawings and built infrastructure. They noted that the method provided significant support towards creating a digital twin of railway infrastructure.

Dzhusupova et al. [86] proposed a YOLOv4 [22] based model to detect specific combinations of shapes in P&IDs that represented engineering errors. Domain experts manually labelled 2253 industry P&IDs with eight classes of equipment combinations. A balanced dataset was obtained by creating new examples of rare symbol instances manually. The authors reported around 70% correct recognition, however the results per class were not presented.

The literature shows that deep learning has been employed for various digitisation applications. Amongst the different types of complex EDs and documents used, there was considerable research attention on P&IDs. Diagrams were sourced from a range of industries such as nuclear [3], construction [52], and oil and gas [1]. In addition to digitising drawing elements, existing literature showed that deep learning was also used for related drawing analysis purposes. These include creating a diagram search tool

[81], determining the appropriate manufacturing method [82] and detecting engineering errors [86]. Data contained within EDs is of critical importance, and there is potential for deep learning to be used for additional digitisation applications.

2.2.2 Metrics

Evaluation metrics are calculated using model predictions and the ground truth. The precision, recall and F1 score are calculated using True Positives (TP), False Positives (FP) and False Negative (FN) detections. Precision is the ratio of TP to the number of predicted positives, refer to Equation 2.1. Recall is the ratio of TP to the number of actual positives, refer to Equation 2.2. The F1 score combines the previous two metrics and is defined as the harmonic mean of precision and recall, as shown in Equation 2.3.

$$Precision = \frac{TP}{TP \ + \ FP} \tag{2.1}$$

$$Recall = \frac{TP}{TP \ + \ FN} \tag{2.2}$$

$$F1 \ score = \frac{1}{\frac{1}{2}(\frac{1}{Precision} + \frac{1}{Recall})} \tag{2.3}$$

A TP detection is defined using object class and location. Firstly, the predicted symbol class must match that of the ground truth. Secondly, the Intersection Over Union (IOU) (Equation 2.4) is considered.

$$IOU = \frac{Area \ of \ Overlap}{Area \ of \ Union} \tag{2.4}$$

Symbol detection methods were also commonly evaluated using the mean Average Precision (mAP). This is defined as the mean of the Average Precision (AP) across all classes, as shown in Equation 2.5. Here AP_i is the AP of the i-th class and C is the total number of classes.

$$mAP = \frac{1}{C}\sum_{i=1}^{C} AP_i \tag{2.5}$$

The AP for each class is defined as the Area Under the Curve (AUC) of the precision-recall curve. This metric is commonly specified at an IOU threshold of 0.5. Note that other IOU thresholds may be specified, for example the Common Objects in Context

(COCO) dataset [87] uses $AP@[.5 : .05 : .95]$, which calculates the average AP at ten different IOU thresholds.

2.2.3 Symbols

Symbols are considered one of the main drawing elements in EDs. Examples of symbols are shown in Figure 2.3. Symbol recognition can be a complex task for multiple reasons. Each diagram typically contains numerous symbol instances, for example, one study reported on average 180 symbols per P&ID [1]. Symbols represent a wide range of equipment types, and consequently, they vary in size and shape. Additionally, there is often a low amount of interclass variation [43, 34] which can result in difficulty distinguishing between symbol classes, refer to Figure 2.4. Moreover, symbols may be overlapped by other drawing elements [11], shown in varying orientations [11], represented by simple shapes [60] or even by only a few lines [61].

Figure 2.3: Examples of engineering symbols as shown in the diagram legend

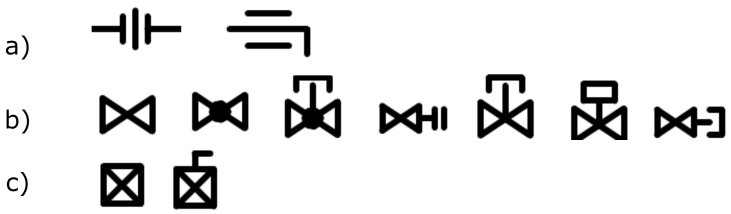


Figure 2.4: Visually similar symbols from mechanical EDs a) Union and Butterfly Valve, b) Gate Valve, Globe Valve, Lockable Flow Control Valve, Hose-End Drain Valve, Lockshield Valve, Automatic Control Valve, Valve and Capped Provision, c) Flow Switch and Balancing Valve (Plug)

Recent literature shows an increasing number of deep learning-based methods for recognising symbols in EDs, as shown in Table 2.2. The most commonly used methods were object detection models. These models predict the location, defined by a bounding box, and the class of objects within an image.

Faster R-CNN [88] based methods were popular for engineering symbol detection [60, 63, 3, 57, 67, 4, 71, 2]. Faster R-CNN is a two-stage object detector presented in 2015. Two related models were published earlier [80, 93]. Region-based CNN (R-CNN) [80] was created in 2014. The selective search algorithm [94] was used to generate

Table 2.2: Symbol recognition methods seen in the literature on diagram digitisation

Model Type	**Model based on**	**Reference**
Detection	YOLOv1 [19]	Zhao et al. [52]
	YOLOv2 [20]	Rezvanifar et al. [61], Gupta et al. [72]
	YOLOv3 [21]	Elyan et al. [1], Nurminen et al. [11]
	YOLOv5 [23]	Toral et al. [58], Hantach et al. [45], Haar et al. [75]
	Faster R-CNN [88]	Ziran and Marinai [60], Nguyen et al. [63], Gao et al. [3], Stinner et al. [57], Hu et al. [67], Joy and Mounsef [4], Sarkar et al. [71], Jakubik et al. [2], Theisen et al. [29]
	R-FCN [78]	Gao et al. [3]
Classification	CNN	Mani et al. [44], Yu et al. [51]
	TBMSL-Net [89]	Paliwal et al. [43]
Segmentation	FCN [90]	Rahul et al. [34], Paliwal et al. [43]
	Mask R-CNN [91]	Bickel et al. [70], Bickel et al. [73]
Graph	DGCNN [92]	Paliwal et al. [6]
	GNN	Renton et al. [62], Renton et al. [66]

around 2,000 region proposals from the input image. CNN features were extracted from each region. These features were then input into class-specific linear SVMs for classification purposes. On the prominent PASCAL Visual Object Classes (VOC) [95] dataset, 30% relative improvement was reported over traditional methods based on features such as HOG [47]. However, the method was computationally slow. Separate CNN computation was required for each region proposal. Fast Region-based CNN (Fast R-CNN) [93] was presented the following year. The model was designed to speed up computation compared to R-CNN. One convolutional feature map was produced for the whole input image. Then, a feature vector was extracted for each region using a Regions of Interest (ROI) pooling layer. Class probabilities and bounding box positions were predicted for each region. Later that same year, Faster R-CNN [88] was proposed. A Region Proposal Network (RPN) was introduced to speed up the costly region proposal. Convolutional features were shared between the RPN and the downstream CNN.

The feature extraction network used in Faster R-CNN was changed in several studies [3, 78, 67]. For example, Gao et al. [3] developed a Faster R-CNN component detection method. A dataset of 68 nuclear power plant diagrams was used. Components were split into three groups based on aspect ratio and scaling factor. These groups were small symbols, steam generator symbols and pipes. A separate model was trained for each group. ResNet-50 [96] was used as the feature extractor. ResNet-50 is a type of residual network with 50 layers. The mAP was 96.6%, 98% and 92% for each group.

Two other models were evaluated for the detection of the small symbols. The first was Faster R-CNN with Inception [97] network. Although 100% AP was still obtained for certain classes, lower performance was observed overall. A R-FCN model [78] with ResNet-50 was also evaluated. Dai et al. [78] introduced R-FCN in 2016. All trainable layers in R-FCN are convolutional. Faster inference time was reported compared to Faster R-CNN [78]. Although the authors of [78] reported comparative performance to Faster R-CNN on the PASCAL VOC dataset [98], this was not the case on the nuclear power plant diagrams. The reported AP was significantly lower at 16.24%. The authors used publicly available diagrams, which may be simplified compared to those in a real-world scenario.

Hu et al. [67] presented an approach to detect the surface roughness symbol from mechanical drawings. A dataset of 3612 mechanical drawings was used. The approach involved symbol detection and text detection. Various object detection models were evaluated. The highest recall and F1 score were reported with Faster R-CNN using ResNet-101 [96] in surface roughness detection. The authors used Single Shot Detector (SSD) [99] with ResNet-50 for localising text and LeNet [100] for character recognition. An F1 score of 96% was reported. The approach was designed specifically for the surface roughness symbol and may be limited in applicability to a wider range of symbols.

Several ED studies required the use of a diagram legend [4, 71]. For example, Joy and Mounsef [4] used a Faster R-CNN method with ResNet-50 for symbol detection in electrical engineering diagrams. First, symbol shapes were obtained using morphological operations to identify symbol grid cells in the legend table. Next, data augmentation was used to increase the available training data. Detection and recognition rates of 83% and above were reported on a small test set of five diagrams. Increasing the training data diversity may help to improve the results. Sarkar et al. [71] also used a Faster R-CNN model for symbol detection in EDs. All symbols were treated as belonging to one class. Detected symbols were then assigned a class based on similarity with the symbols in the diagram legend. Two similarity measures were evaluated. The first was based on traditional SIFT [48] features. The second employed a CNN as a feature extractor. Better performance was reported using the SIFT-based approach. These studies relied on the use of a diagram legend, however, this may not be available in practice. Moreover, symbols can be present in the diagrams that do not appear in the legend [71].

Yun et al. [101] also created an R-CNN-based method for symbol recognition from P&IDs. Ten industry P&IDs were used. Region proposals were generated using image processing methods customised for each symbol type. Positive and negative regions

were obtained. The negative regions were divided into classes using negative class decomposition through unsupervised learning models, namely k-means and Deep Adaptive image Clustering (DAC) [102]. Positive regions were assigned classes manually. Results showed that the incorporation of the negative classes reduced false positives. A slight improvement was reported using DAC compared to k-means. This method is rule-based and requires manual adjustment for a different use case.

Faster R-CNN based symbol detection methods were also used on floor plan images [60, 2]. For instance, Ziran and Marinai [60] presented a Faster R-CNN method for object detection in floor plan images. Two datasets were used. The first contained 135 diverse floor plans obtained from internet search queries. The second consisted of 160 industry floor plans sourced from an architectural firm. Although detailed results of the preliminary experiments were unavailable, improved performance using Faster R-CNN compared to SSD was reported. The initial performance on the first dataset was comparatively low, at 0.26 mAP. Data augmentation and anchor specification increased the mAP to 0.31. For the second, more standardised dataset, the mAP was higher at 0.86. Additionally, the authors used transfer learning to improve performance on the more diverse dataset. The model was pre-trained on the second dataset and then fine-tuned on the first dataset. Performance improved by 0.08 mAP.

Jakubik et al. [2] presented a human-in-the-loop system for object detection and classification in floor plans. The symbol detection method was based on Faster R-CNN. A training dataset of $20,000$ synthetic images was created using legend symbols and data augmentation. The test set of 44 industry floor plans was manually annotated with 5907 symbols from 39 classes. An uncertainty score was calculated for each detected and then classified symbol. Symbols were then labelled by a human expert in order of decreasing uncertainty. A range of uncertainty measures was evaluated. Increased accuracy was reported compared to random selection at 50% of the labelling budget, using all but one uncertainty measure.

One-stage object detection models have also been used for engineering symbol detection [52, 61, 1, 58, 103]. These models are faster than two-stage models. One of the most well-known one-stage object detection models is YOLO [19], which was created in 2016. A real-time inference speed of 45 frames per second (fps) was reported. In contrast, the authors of Faster R-CNN [88] reported a lower processing speed of 5 fps. YOLO is comparatively faster as a single neural network was used to predict bounding boxes and class probabilities. The network had 24 convolutional layers followed by 2 fully connected layers. The input image is divided into a S x S grid. Objects are assigned to the grid cell that contains the object centre. Each grid cell predicts B bounding boxes. The centre of the bounding box is defined relative to the grid cell, whereas the

width and height are predicted relative to the whole image. Class-specific confidence scores for each box are also predicted. Several extensions to the initial YOLO version [19] were proposed. YOLOv2 [20] contained several modifications, including multi-scale training and anchor boxes. The base network, Darknet-19, had 19 convolutional layers. In YOLOv3 [21], the bounding boxes were predicted at three different scales. A feature extractor with 53 convolutional layers was used. Newer versions, YOLOv4 [22], YOLOv5 [23], YOLOv6 [24], YOLOv7 [25], YOLOv8 [26], YOLOv9 [104] and YOLOv10 [105] were also proposed. Another one-stage object detection model is SSD [99]. The single network employs multi-scale feature maps for predictions. RetinaNet [106] is also a one-stage detector. The model was introduced in 2017 and employs the novel focal loss function.

YOLO-based methods have been used for symbol detection in several different drawing types, including structural diagrams [52], floor plans [61], and P&IDs [1]. For example, Zhao et al. [52] presented a YOLO-based method to detect components in scanned structural diagrams. Five symbol classes were considered. Related semantic information, such as the symbol tag, was included in the symbol bounding box. Data augmentation increased the dataset size from 500 to 1500 images. F1 score of 86.7% and above was reported.

Focussing on architectural floor plans, Rezvanifar et al. [61] proposed a YOLOv2 symbol detection method. A private dataset of 115 diagrams was used. Various backbone networks were evaluated. Higher mAP was reported using ResNet-50 compared to Darknet-19 and Xception [107]. However, detection performance varied widely across the 12 classes considered. For example, the accuracy for the window symbol was 76% compared to 100% for the shower symbol. This may be due to the window symbol's varying aspect ratio and visual similarity compared to other image components. Additionally, 70 floor plans from the public Systems Evaluation SYnthetic Documents (SESYD) dataset were used. Results improved compared to traditional symbol spotting methods. However, the authors observed that the SESYD diagrams were simpler than typical real-world floor plans. Moreover, there were no intra-class symbol variations. Although YOLOv3 performance was not evaluated, its multi-scale prediction may improve the performance on the relatively small symbols [21].

In another study, Elyan et al. [1] created methods for symbol detection and classification in P&IDs. A dataset of 172 industry P&IDs from an oil and gas company was used. The symbol detection method was based on YOLOv3. Accuracy was 95% across 25 symbol classes. The authors observed lower class accuracy for the least represented classes. Additionally, a Deep Generative Adversarial Neural Network was presented to handle class imbalance for symbol classification. GANs [108] are deep learning models

designed to generate data. GANs contain two models. These are a generator and a discriminator. A generative model is trained to produce fake data which is indistinguishable from real data by the discriminator. The authors used a Multiple Fake Class GAN (MFC-GAN) [77] to generate synthetic instances of the minority class. Experiments showed that realistic synthetic samples were generated. The synthetic instances improved CNN classification. Note that these results were based on using only a few training samples per class. For instance, the Angle Choke Valve class was represented by only two instances in the initial dataset.

A number of researchers used a CNN classifier with a sliding window approach to detect symbols in engineering diagrams [44, 51]. Classifiers predict an object class for a given image. For instance, Mani et al. [44] created a classification-based method for extracting two symbol classes from P&IDs. A dataset of 29 P&IDs was used. The sliding window method extracted fixed-size image patches from the diagram. The CNN had three convolutional layers and two fully connected layers. Patches were classified as 'tag', 'Locally Mounted Instrument' (LMI) or 'no symbol'. On 11 test diagrams, tags were classified with a precision of 100% and recall of 98%. LMIs were classified with a precision of 85% and recall of 95%. According to the authors, results were poorer for LMIs due to visually similar components.

Yu et al. [51] used a similar approach to detect symbols in P&IDs. A dataset of 70 industry P&IDs was used. First, image processing techniques were employed for diagram realignment and to remove the outer border. An AlexNet [18] classifier was then used with a sliding window approach. Candidate symbol regions were identified by means of morphological close and open operations. The window size was customised for each symbol class. The symbol recognition accuracy was 91.6%. This method was tested on a limited test set of only two P&IDs. Moreover, the test diagrams contained a simple equipment layout with little interference between components. Whilst promising results were reported in these studies, this method would likely become computationally expensive for a more extensive use case. Although the sliding window approach was frequently used with traditional methods, including Haar cascades [109] and Deformable Part Models [110], there is a prohibitive computational cost of classifying each window using a CNN. Moreover, small stride and multi-scale windows are typically required to obtain high localisation accuracy.

Segmentation-based methods have also been used to digitise symbols from engineering diagrams [43, 34]. Rather than predicting a symbol bounding box, segmentation methods generate pixel-level predictions. For instance, Rahul et al. [34] created a Fully Convolutional Network (FCN) [90] method to segment 10 symbol classes from P&IDs. The authors used four real-world P&IDs from an oil company. F1 scores of 0.87 and

above were recorded. However, the authors reported that their methods' performance dropped in the presence of visually similar symbols. This was observed in a dataset of P&IDs with a relatively blank background.

Paliwal et al. [43] used a combination of methods to recognise symbols in P&IDs. Basic shape symbols were detected using traditional methods, such as Hough transform for circle detection. Complex symbols were localised using an FCN [90] segmentation model and classified using Three-Branch and Multi-Scale Learning Network (TBMSL-Net) [89]. The methods were evaluated on 100 synthetic P&IDs and a smaller private dataset of 12 real-world P&IDs. An F1 score of 0.820 and above across 32 symbol classes was reported on the synthetic test set. Improved performance compared to Rahul et al. [34] was observed on the real-world P&IDs. The use of the Hough transform for basic shapes is unlikely to generalise well across different symbol sizes and appearance variations.

Graph-based methods have been used to recognise symbols in EDs [6, 66, 62]. A graph in this context is comprised of nodes connected by edges. For example, Paliwal et al. [6] created a Dynamic Graph Convolutional Neural Network (DGCNN) [92] to recognise symbols in P&IDs. The symbols were represented in graph form and then classified using the DGCNN. Classification accuracy of 86% was recorded on 100 synthetic P&IDs. Symbol misclassifications were observed due to noise and clutter. The method was compared to the FCN based-method presented by Rahul et al. [34] on 12 real-world P&IDs, and improved F1 scores were reported for 3 out of 11 classes. Only one instance per class was used to train the DGCNN. To increase the model's robustness, it was augmented with embeddings from a ResNet-34 network pre-trained on symbols.

Renton et al. [66] introduced a GNN method for symbol detection and classification in floor plans. A dataset of 200 floor plans was used. First, the floor plans were converted into Region Adjacency Graphs (RAGs). The nodes represented parts of images, and the edges represented relationships between these parts. Using a GNN, nodes were classified as one of 17 symbol types. This work was developed further in Renton et al. [62], when the authors clustered the nodes into subgraphs corresponding to symbols. Here a symbol detection accuracy of 86% was reported.

Mizanur Rahman et al. [69] employed a combination of graph-based methods and Faster R-CNN for symbol detection in circuit diagrams. A dataset of 218 diagrams was used. The symbol detection method was based on Faster R-CNN with ResNet-50. Graph methods were then used to refine the model. Detected symbols were graph nodes. Symbol-to-symbol connectors, identified through image processing-based blob

detection, were graph edges. Graph Convolutional Networks (GCN) and node degree comparison were used to identify graph anomalies, which were potentially false negative predictions from Faster R-CNN. The Faster R-CNN model was then fine-tuned using the anomaly regions. An improvement in recall between 2 and 4% was reported, although the overall F1 score decreased by up to 3%. Additionally, graph refinement techniques were used to identify incorrectly labelled nodes. However, the recall was reduced by up to 3% compared to Faster R-CNN alone. One drawback of the symbol-to-symbol connection method was that it missed complex connections which looped around a symbol.

Studies on engineering symbols classification are also available in the published literature [31, 111]. For example, Elyan et al. [111] presented work on engineering symbols classification. Symbols were classified using Random Forest (RF), SVM and CNN. Comparable results with all three methods were reported. The authors also applied a clustering-based approach to find within-class similarities. This benefitted RF and SVM performance. However, there was a slight decrease in CNN performance, potentially due to the limited dataset size.

In summary, it can be said that despite the use of state-of-the-art deep learning methods, detecting and recognising symbols in complex documents and EDs continues to be an inherently challenging problem. Many factors contribute to the challenge including symbol characteristics such as a lack of features [60, 61], high intra-class variation [61] and low inter-class variation [43, 34]. Moreover, the lack of publicly available annotated datasets [13] increases the difficulty of the task. Consequently, further research is required to improve methods for symbol digitisation from complex EDs.

2.2.4 Text

Text is another major component that exists in almost all types of EDs. Text digitisation here involves two stages, first, the detection of the text and second, the recognition of the text. This is illustrated in Figure 2.5. Both the detection and recognition steps are considered challenging for multiple reasons. Each diagram typically contains numerous text strings. For example, Jamieson et al. [37] used 172 P&IDs and reported on average 415 text instances per diagram, whilst Francois et al. [65] used 330 engineering documents and reported on average 440 text boxes. Unlike text in documents with a specific format, text in complex diagrams can be present anywhere in the drawing [65], including within symbols [44]. Additionally, these text strings are often shown in various fonts [34], printed in multiple orientations [37, 3, 58] and vary widely in length [65]. Moreover, this text is often present in a cluttered environment and can overlap other diagram elements [8], as is shown in Figure 2.6.

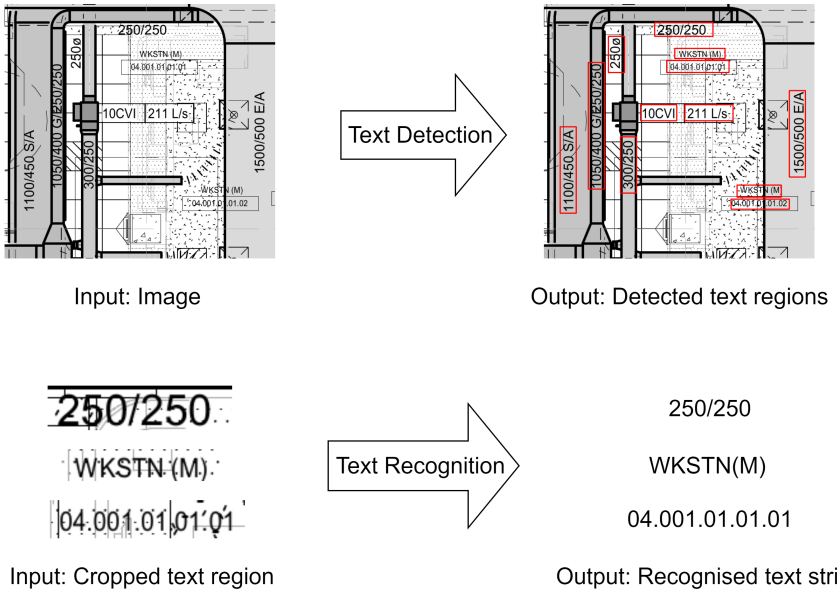


Figure 2.5: Text digitisation is most commonly approached in recent ED literature in two steps. Firstly, a text detection model predicts text regions within an image. Secondly, a text recognition model predicts a text string from a cropped text instance

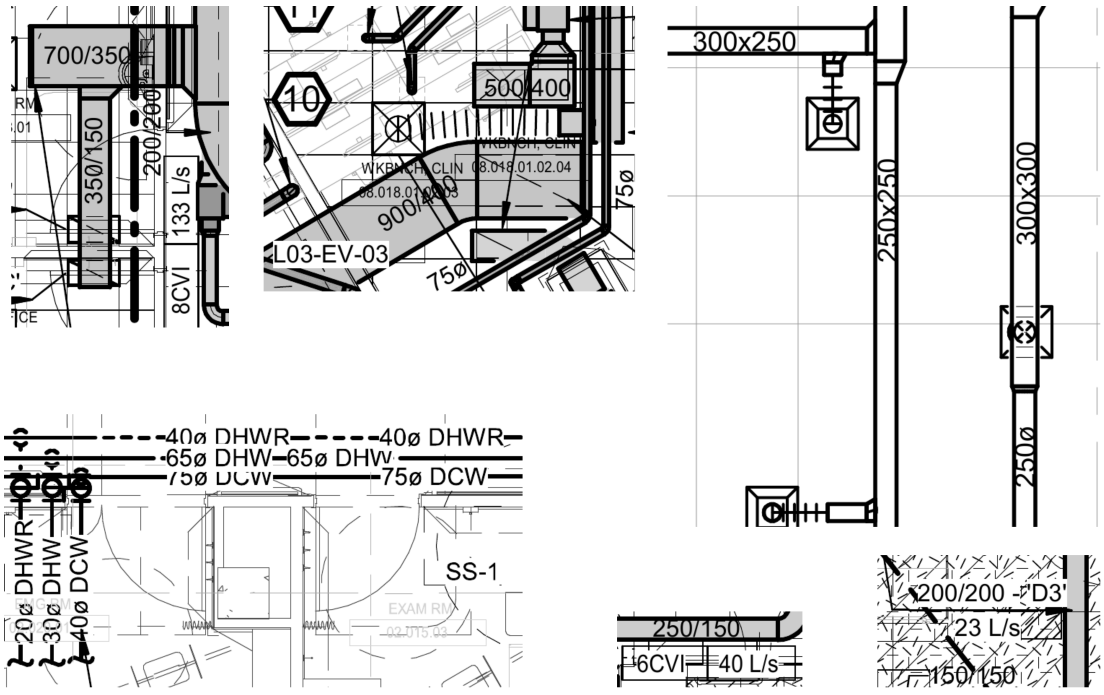


Figure 2.6: Text within EDs is commonly shown in multiple orientations, a cluttered environment and overlapped by separate text strings or other shapes

Whilst there has been a considerable amount of research on text digitisation, most of it was focused on scene text [112]. Scene text is defined as text that appears in natural environments [113, 114]. However, text in undigitised complex documents presents unique challenges that are generally not observed for text in natural scenes. These specific challenges include image degradation [13] and the presence of multiple visually similar drawing elements. Complex documents often lack colour features that can be used to distinguish text from the background. Moreover, the task is more complicated than digitising text from standard format documents, where text is typically presented in straight lines and composed of known words.

There is a clear shift toward using deep learning-based methods in text digitisation, as shown in a relatively recent extensive review paper [113]. Deep learning models automatically extract image features, whereas traditional text methods rely heavily on manually extracted features. For instance, text detection methods commonly used image features based on colour, edge, stroke and texture [112]. Specific features used included HOG, Stroke Width Transform (SWT), and Maximally Stable Extremal Regions (MSER). Two popular traditional text detection methods were based on Connected Component Analysis (CCA) and sliding window classification [112, 113]. CCA methods extract candidate text components and then filter out non-text regions using heuristic or feature-based methods [113].

Table 2.3: Text digitisation methods used in the related literature

Step	**Type**	**Method**	**Reference**
Detection	Text	EAST [115]	Mani et al. [44], Jamieson et al. [37], Francois et al. [65]
		CTPN [116]	Rahul et al. [34], Yu et al. [51]
		SegLink [79]	Gao et al. [3]
		CRAFT [117]	Paliwal et al. [43], Kim et al. [32] Mafipour et al. [74]
	Object	Faster R-CNN [88]	Nguyen et al. [63], Rumalshan et al. [76]
		SSD [99]	Hu et al. [67], Rumalshan et al. [76]
		YOLOv3 [21]	Rumalshan et al. [76]
		YOLOv5 [23]	Toral et al. [58]
Recognition		Tesseract	Rahul et al. [34], Sinha et al. [54], Kang et al. [8], Mani et al. [44], Jamieson et al. [37], Paliwal et al. [43], Kim et al. [32], Joy and Mounsef [4], Francois et al. [65], Jakubik et al. [2], Toral et al. [58]
		CNN	Nguyen et al. [63]

Various deep learning models were used to detect text in complex diagrams, as shown in Table 2.3. The majority of studies used models designed for text detection, including Character Region Awareness for Text detection (CRAFT) [117], EAST [115], Connectionist Text Proposal Network (CTPN) [116] and SegLink [79]. CRAFT [117] was designed to localise individual characters, whereas EAST [115] uses a FCN to predict word or text line instances from full images. Meanwhile, CTPN [116] localises text lines, while SegLink [79] decomposes text into oriented boxes (segments) connected by links.

Object detection models have also been used to detect text in EDs [63, 67, 58]. For example, Nguyen et al. [63] created a Faster R-CNN method to detect symbols and text in scanned technical diagrams. A large dataset of 4630 technical diagrams was used. Five classes were considered. Individual characters were recognised from the text regions using a CNN separation line classifier and a CNN character classifier. The average F1 score was 89%, although performance varied across object classes. The lowest F1 score, 78%, was reported for the least represented class. Text recognition exact match accuracy was 68.5%. Toral et al. [58] also used an object detection model for text detection. They created a YOLOv5 method to detect pipe specifications and connection points. Pipe specifications are text strings with a specific format, whereas the connection point symbol contains a short text string. A heuristic method was applied to the detected object regions to obtain text regions. The text was recognised using Tesseract. Detection and recognition accuracy of 93% and 94% was reported. Rumalshan et al. [76] presented methods for component detection in railway technical maps. The components were a combination of text codes and simple shapes. Their Faster R-CNN method outperformed YOLOv3 and SSD methods. Seeded region growing [118] was used to pre-process the detected regions prior to Optical Character Recognition (OCR). White pixels at the edge of the regions were the seeds.

Whilst there is a range of deep learning models designed for text recognition, a popular choice was to use Tesseract software [119], as shown in Table 2.3. The latest versions of this employ deep learning. Deep learning text recognition models can be considered segmentation-based or segmentation-free methods [120]. Segmentation methods generally contain pre-processing, character segmentation and character recognition steps. In contrast, segmentation-free approaches predict a text string from the entire text instance. For example, these methods may comprise image pre-processing, feature extraction, sequence modelling, and prediction steps [120]. Sequence modelling considers contextual information within a character sequence. A type of Recurrent Neural Network (RNN) known as a Bi-directional LSTM Network is often used. The two main prediction methods are attention based [121] and Connectionist Temporal Classification

(CTC) [122]. One example of a deep learning text recognition method is the Convolutional Recurrent Neural Network (CRNN) [123]. It combines a CNN, an RNN and a transcription layer.

EDs may contain symbols and shapes that are visually similar to text. This was reported in a study by Jamieson et al. [37]. Here, the authors built a framework to digitise EDs. They used EAST [115] to localise text and LSTM-based Tesseract [119] for text recognition. Good performance was achieved overall with 90% of text instances detected. However, false positives were predicted for shapes visually similar to text, including dashed lines and symbol sections. Yu et al. [51] also reported a similar challenge. They used a CTPN [116] based method to detect text in P&IDs. Character recognition accuracy was 83.1%. Although the two test diagrams used had a simple equipment layout, part of a symbol was recognised as a character.

Another challenging problem with text digitisation is the orientation of the text. This was reported in several studies [32, 3, 43], and various methods were proposed to handle it. For example, Kim et al. [32] created methods to recognise symbols and text in P&IDs. The text was detected using the easyOCR[1] framework and recognised using Tesseract [119]. EasyOCR is based on CRAFT [117] and CRNN methods. Text rotation was estimated based on aspect ratio and text recognition score. Text detection and recognition combined precision and recall were 0.94 and 0.92, respectively. The authors used P&IDs that contained no noise or transformations, however this is not necessarily the case in practice [27]. Text digitisation methods were also applied on rotated diagrams [3, 43]. For instance, Paliwal et al. [43] proposed methods to digitise P&IDs. First, the text was detected using CRAFT and recognised using Tesseract. Then, the diagram was rotated and the process was repeated to capture missing vertical text strings. Text detection and recognition accuracy of 87.18% and 79.21% was reported.

Another key challenge is that text in EDs is often composed of codes rather than known words. This differs from the text in other document types, which typically belongs to a specific lexicon. Rahul et al. [34] used prior knowledge of the text structure when they digitised pipeline codes from P&IDs. The method was based on a CTPN model [116] and Tesseract. Text detection accuracy was 90%. The pipeline codes had a fixed structure, which was used to filter out false positive text strings. However, complex diagrams contain text for numerous reasons, and details of the various structures are not always available.

Francois et al. [65] proposed a correction method for recognised text. The dataset

[1] https://github.com/JaidedAI/EasyOCR/

comprised 330 industry engineering documents, including P&IDs and isometrics. Their text method was based on the EAST model [115] and Tesseract. A post-OCR correction step involved text clustering using affinity propagation. The Levenshtein distance was used as the similarity measure. Clusters were defined to maximise the similarity score between data points. The post-OCR correction improved tag recognition from 75% to 82%. However, the application of this method to other scenarios relies on the text character structure being known in advance.

Text digitisation from complex EDs remains challenging. Although text detection and recognition has received large research interest [113, 112, 120], the majority was focussed on scene text [112]. The literature shows that text within EDs presents different challenges. This is because the text can be present anywhere in the image [65], of multiple orientations [37], and is frequently overlapped by other shapes. One particular challenge for deep learning models is distinguishing text from other similar shapes in the diagram [37, 51]. Moreover, compared to other domains, there is a lack of publicly available annotated text datasets. Further research is necessary to enable accurate text detection and recognition from complex EDs.

2.2.5 Connectors

Connectors in EDs represent the relationship between symbols. The simplest representation of a connector is a solid line, which typically represents a pipeline. More complex line types such as dotted lines and dashed lines are also used, which represent specialised connectors such as electrical signal or air lines. Examples of different connectors can be seen in Figure 2.7. Although connector extraction may seem a simple task, it can be difficult for computer vision methods to distinguish between connectors and other shapes in the diagram. This problem occurs as all diagram elements are essentially composed of lines. For instance, the character 'l' may also be considered a short line. Methods to overcome this challenge and accurately digitise connectors are required, as their information is vital for understanding the flow through a system.

Despite the recent advances in deep learning, methods employed for line detection are still primarily based on traditional approaches [34, 57, 51, 8]. For instance, Yu et al. [51] introduced methods for line recognition in P&IDs. First, image processing techniques were employed for diagram realignment and to remove the outer border. A series of image processing methods was used for line recognition. This involved determining the most common line thickness. Reported accuracy was 90.6%. The authors reported that symbol sections were recognised as lines. Difficulty in recognising dotted and diagonal lines was also reported in this study. This was observed even in a very limited test set of only two P&IDs which contained a simple equipment layout with little interference

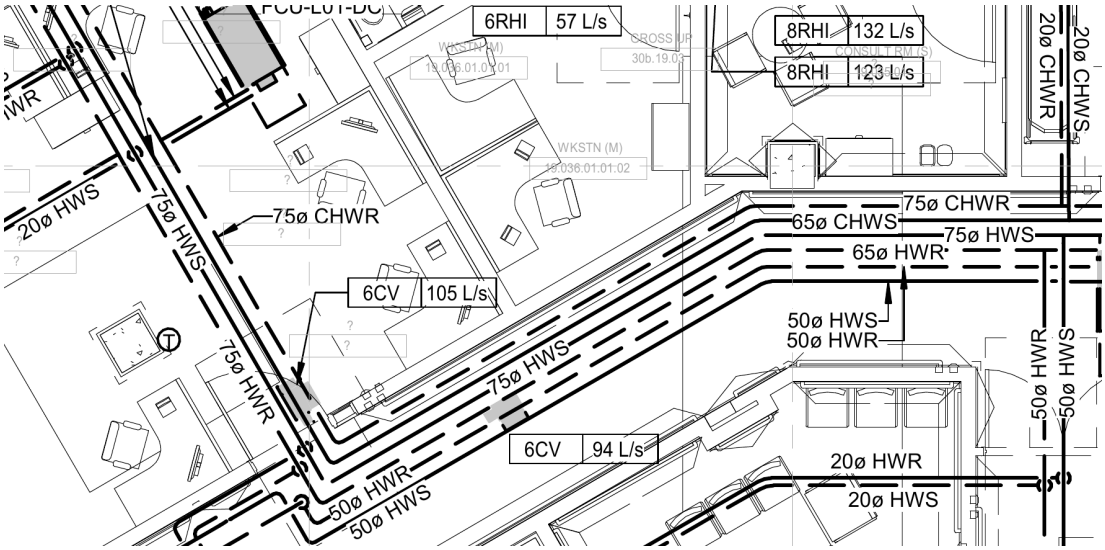


Figure 2.7: Section of ED showing different line representations

between components. Kang et al. [8] also used a traditional method for line extraction from P&IDs. Lines were extracted based on the symbol connection point and sliding window method. Particular difficulties recognising diagonal and separated lines were reported.

Other traditional line extraction methods include those based on the Hough transform or kernels. In a study by Stinner et al. [57], lines were detected using binarisation and Hough transform. Line crossings were detected using a line intersection algorithm. Meanwhile, Rahul et al. [34] used the more efficient Probabilistic Hough Transform [124] to detect pipelines in P&IDs. Although the P&IDs appear to have a relatively blank background, the pipeline detection accuracy, 65%, was still effected by noise and overlapping drawing elements. In the kernel-based method, a small filter is passed over the diagram and a convolution operation is applied. Paliwal et al. [43] used a kernel-based method to detect lines in P&IDs. A higher detection accuracy for complete lines (99%) than for dashed lines (83%) was reported. The authors considered the line width and image spatial resolution when designing the structuring element matrix. It should be noted, however, that kernel-based methods are very sensitive to noise and the thickness of lines.

Although not commonly seen in the literature, line detection may be considered as an object detection problem. This approach was employed by Moon et al. [56] in their study on line detection in P&IDs. A dataset of 82 remodelled industry P&IDs was used. First, the P&ID border was removed using binarisation, pixel processing and morphological operations. A RetinaNet [106] object detection model was used to detect flow arrows and specialised line types, such as electrical signal lines. These lines

were composed of either a line with a shape overlaid, or a series of dashes. In the latter case, each dash was treated as an object. A post-processing step was needed to merge the detected line sections. Continuous lines were detected using traditional image processing methods, including line thinning and Hough transform. Symbol and text regions detected using the method created by Kim et al. [32] were removed to discard false-positive lines. A precision of 96.1% and recall of 89.6% was reported. The dataset was imbalanced, although the results showed that highest performance was not always obtained for the most represented class.

Connector detection is also considered a challenging problem. Despite the recent popularity of deep learning digitisation methods for symbols and text, this is not the case for connector digitisation methods. Methods used for this task are still primarily based on traditional approaches [34, 8, 57]. Such approaches include the Hough transform, Probabilistic Hough Transform [124] and kernel-based methods. Furthermore, the scale of the problem is increased as multiple line types can be present in one diagram [56, 34, 8]. Distinguishing connectors from other shapes in the diagram can be difficult for computer vision methods. Moreover, there is a lack of connector-labelled datasets for use with deep learning models. Therefore, accurate connector detection from complex EDs remains difficult, and improved methods are required.

2.3 Challenges

Although there are numerous benefits of using deep learning methods for diagram digitisation, such as their generalisability to the variations seen in the drawings and automatic feature extraction, the existing literature also suggests various challenges. These are a lack of public datasets, data annotation, evaluation, class imbalance and contextualisation. Compared to traditional methods, deep learning methods typically require large quantities of training data. Due to proprietary and confidentiality reasons, diagram datasets are generally not available in the public domain. Furthermore, when datasets can be obtained, they typically need to be labelled for use with supervised deep learning models. The lack of annotated datasets increases the difficulty of evaluating digitisation methods. Another challenge arises from the fact that while deep learning models are typically designed for balanced datasets, ED datasets are inherently imbalanced. A detailed discussion of these challenges is presented in this section.

2.3.1 Datasets

The lack of publicly available ED datasets makes it difficult to compare and benchmark various methods. As can be seen in Table 2.4, most methods are evaluated using proprietary datasets. It should also be pointed out that there is a vast variety of

formats for these drawings. Specific organisations or even specific projects may adopt their own drawing formats, which would not be captured in publicly available datasets. This means that retraining models to suit specific ED datasets is an important and necessary factor to consider. One example of a public dataset used in the digitisation literature is the SESYD floor plan dataset, which contains 1000 images [61]. However, this dataset is synthetic, contained no intra-class symbol variations and was considered simpler than typical real-world floor plans [61]. Moreover, researchers working on floor plan digitisation still report a lack of available training data [60].

Table 2.4: Datasets in relevant literature

Reference	Year	Diagram Type	Number	Source
Ziran and Marinai [60]	2018	floor plans	135 & 160	public & industry
Rahul et al. [34]	2019	P&IDs	4	industry
Sinha et al. [54]	2019	P&IDs	106	private
Yu et al. [51]	2019	P&IDs	70	industry
Kang et al. [8]	2019	P&IDs	3	-
Renton et al. [66]	2019	floor plans	200	-
Mani et al. [44]	2020	P&IDs	29	-
Gao et al. [3]	2020	nuclear power plant diagram	68	public
Elyan et al. [1]	2020	P&IDs	172	industry
Zhao et al. [52]	2020	structural drawings	500	private
Rezvanifar et al. [61]	2020	architectural drawings & floor plans	115 & 70	industry & public
Moreno-Garcia et al. [55]	2020	P&IDs	8	industry
Jamieson et al. [37]	2020	P&IDs	172	industry
Nurminen et al. [11]	2020	P&IDs	22000	synthetic & industry
Paliwal et al. [43]	2021	P&IDs	100 & 12	synthetic & industry
Moon et al. [56]	2021	P&IDs	82	remodelled from industry
Nguyen et al. [63]	2021	technical diagrams	4630	real world
Kim et al. [32]	2021	P&IDs	82	remodelled from industry
Stinner et al. [57]	2021	P&IDs & diagrams	5 & 13 & 84	industry & public
Paliwal et al. [6]	2021	P&IDs	100 & 12	synthetic & industry
Hu et al. [67]	2021	mechanical drawings	3612	-
Joy and Mounsef [4]	2021	electrical plans	5	-
Schiebel et al. [68]	2021	EDs	7	public & industry
Kim et al. [42]	2021	floor plans	230	private
Renton et al. [62]	2021	floor plans	200	-
Toral et al. [58]	2021	P&IDs	85	industry
Hantach et al. [45]	2021	P&IDs	8	real-world
Mizanur Rahman et al. [69]	2021	circuit diagrams	218	public
Sarkar et al. [71]	2022	EDs	342	-
Francois et al. [65]	2022	engineering documents	330	industry
Jakubik et al. [2]	2022	floor plans	44	industry
Xie et al. [82]	2022	EDs	1692	private
Bin et al. [125]	2022	P&IDs	7	-
Gupta et al. [72]	2022	P&IDs	3	industry
Haar et al. [75]	2023	engineering & manufacturing drawings	15 & 1000	real & synthetic
Rumalshan et al. [76]	2023	railway technical maps	69	-
Theisen et al. [29]	2023	process flow diagrams	1005	various public sources

Synthetic diagrams have been utilised in the absence of sufficient real-world data [43, 126, 11, 75, 70]. For instance, Paliwal et al. [43] generated a dataset comprising 500 annotated synthetic P&IDs. Image noise was added. The dataset contained 32 equally represented symbol classes. However, class imbalance is inherent in real-world P&IDs and can cause models to be biased towards overrepresented classes. Sierla et al. [126]

included data extraction from scanned P&IDs as a step in their methodology for the semi-automatic generation of digital twins. YOLO was used for symbol detection. The authors generated artificial images by placing symbols, without associated text and connectors, from process simulation software on a white background. However, these images were relatively simple and did not present the challenges associated with scanned P&IDs. Similarly, Nurminen et al. [11] created artificial images using process simulation software. They created a YOLOv3-based model for symbol detection in P&IDs. The method was evaluated on artificial images and scanned industrial P&IDs. Meanwhile, Bickel et al. [70, 73] generated synthetic training data for symbol detection in principle sketches. They used a fixed set of rules to generate symbols, which was practical in this case owing to the defined representation limits of the drawings used.

Stinner et al. [57] used images from symbol standards and internet search images to increase the training dataset size. They presented work on extracting symbols, lines and line crossings from P&IDs. The authors used five industry P&IDs. They used a Faster R-CNN-based method to detect four symbol types. The authors reported 93% AP over all symbol classes. However, performance was lower for certain object classes compared to others. They also reported false positive detections. These would potentially be increased as a result of the model training approach, in which a variety of sources including internet search data was used, meaning that the appearance of the training symbols may not exactly match that of the target symbols.

Haar et al. [75] presented symbol and text detection methods for engineering and manufacturing drawings. A dataset of 15 real drawings and 1000 synthetic images was used. Synthetic data was generated by cropping symbols from the real drawings and randomly placing them on the basic drawings with varying orientations and sizes. YOLOv5 was used to detect symbols. EasyOCR was used for the text. The model utilised VGG and ResNet for feature extraction, LSTM and CTC. The YOLOv5 model performance on the real diagrams (36.4 mAP) was lower than on the synthetic dataset (87.6 mAP). The text method was evaluated on five diagrams and correctly recognised 68% of text characters. Mathematical special characters and rotated texts were highlighted as a challenge.

Although there is a lack of text datasets for EDs, many text datasets exist in other domains. In 2015, commonly used text datasets were discussed in a review [112]. The largest dataset mentioned was IIIT5K Word [127], which contains $5,000$ cropped images. Since then, demand for significantly bigger datasets to train deep learning models has increased. Today, the largest text datasets contain millions of synthetic text instances [120]. For example, Synth90K [128] contains 9 million synthetic annotated text instances. The Unreal text dataset [129] comprises 12 million cropped text instances.

In contrast, realistic text datasets are smaller, containing thousands of data samples [120]. Veit et al. [130] introduced the COCO-Text dataset in 2016. The dataset contained over 173k annotated instances of text in natural images, making it the largest dataset of its type at the time. The ICDAR also introduced text datasets [131, 132].

The literature shows an urgent need to have more ED datasets available in the public domain. Most of the proposed digitisation methods were evaluated on proprietary datasets, which may contain a limited number of diagrams [45, 51]. Although synthetic datasets were also used, these diagrams were typically simple in appearance and not as complex as those in the real-world [61, 126]. Public access to diagram datasets would also allow for improved comparison between proposed methods. Therefore, the release of public datasets is crucial to accelerate research and development in the area of ED digitisation.

2.3.2 Data Annotation

Obtaining sufficient annotated data is also regarded as a challenge. When datasets are available, they must be annotated for use with supervised deep learning models. Typically, a large annotated dataset is required for training purposes [2]. Acquiring such data is usually carried out manually. Various software can be used to facilitate this, such as Sloth[2], LabelImg [3] and LabelMe [133]. For example, to obtain a symbol dataset, the user needs to draw a bounding box around the symbol and then label it with the relevant class. These steps are required for every symbol of interest in the diagram. Given the high number of symbols per diagram, the process is very time-consuming, costly and prone to human error. Furthermore, given the technical nature of these drawings, a subject matter expert is normally required to complete this task.

One method to reduce the required labelling effort is to create synthetic training data [3, 125, 72]. The simplest approach is to use traditional image processing algorithms. For instance, Gao et al. [3] presented a method for component detection in nuclear power plant diagrams. They manually annotated symbols and then used traditional data augmentation techniques, such as image resizing, to increase the training symbol instances [3]. The AP increased from 40% to 82% when the training dataset increased from 100 to 1000 images. Gupta et al. [72] created a YOLOv2 method for valve detection in P&IDs. A dataset of three P&IDs was used. Synthetic training data was generated by cropping a symbol and randomly placing it on the background. Experiments showed that model performance improved when the amount of background and similar symbols in the training data was increased. However, evaluation of more than

[2] https://sloth.readthedocs.io/en/latest/
[3] https://github.com/tzutalin/labelImg

one symbol type and one test diagram is required to determine if the method can be applied to other scenarios.

Synthetic training data was also created using generative deep learning models [125, 134]. For example, Bin et al. [125] used a method based on CycleGAN [135] and CNN for P&ID symbol recognition. A dataset of seven P&ID sheets was used. CycleGAN [135] uses unpaired images. The accuracy improved from 90.75% to 92.85% when equal representations of synthetic to authentic samples were used for training. However, the authors reported that the performance gain decreased with a 2:1 ratio of synthetic to authentic samples, as an accuracy of 91.88% was reported. Khallouli et al. [134] presented work on OCR from industrial engineering documents. Nine drawings of ships were used. They used a method based on ScrabbleGAN [136] to generate synthetic word images. The model contains a generator, discriminator and text recogniser. When the synthetic data was added to manually labelled training data, the character recognition accuracy increased from 96.83% to 97.45% and the word recognition accuracy increased from 88.79% to 92.1%. However, it is important to note that for the synthetic images to be of most benefit they need to closely represent the challenges seen in real-world drawings, such as overlapping components and dense representation of equipment.

Most of the relevant literature on ED digitisation used supervised deep learning, which learns from labelled training data. An alternative approach is semi-supervised learning, which uses both labelled and unlabelled data [137]. In contrast, weakly supervised methods use partially labelled data. For example, weakly supervised object detection methods mostly use image-level labels [138]. In the area of scene text detection, Lui et al. [114] presented a semi-supervised method named Semi-Text. ICDAR 2013 [131], ICDAR 2015 [132] and Total-Text [139] datasets were used. A Mask R-CNN based model was pre-trained on the SynthText dataset [140]. Then, positive samples were obtained by applying the model to unannotated images. The model was then retrained using a dataset of positive samples and SynthText data. The performance improved compared to the baseline model.

Data annotation continues to be largely carried out manually, which proved to be extremely time-consuming and costly. Furthermore, as the diagrams are highly technical, identifying the different symbol classes within a diagram typically requires a domain expert. Therefore, improved methods to speed up the data annotation process, or reduce the need for annotated data, are required.

2.3.3 Evaluation

Evaluating deep learning methods for complex document digitisation is considered a complex task. Methods used for symbols, text and connectors must all be evaluated separately. Moreover, multiple different metrics are used for the same task. For instance, symbol digitisation methods are evaluated with various metrics including precision, recall, F1 score and mAP. The lack of standard evaluation protocol, along with the use of disparate datasets, increases the difficulty of thoroughly comparing proposed methods.

Symbol detection methods define a TP at a specific IOU threshold. The PASCAL [95] evaluation metric was often used in the related work [2]. This defines a correct detection if the IOU is over a threshold of 0.5. More stringent criteria to define a correct detection were also seen. For instance, Rezvanifar et al. [61] defined a correct detection if the IOU was over 0.75. Meanwhile, Paliwal et al. [43] defined a correct symbol detection based on an IOU greater than 0.75 and a correct associated text label. Different symbol evaluation metrics may be used in the case of graph-based methods. For example, Renton et al. [62] used a GNN for symbol detection and classification. They defined a correct detection if all the symbol nodes representing a symbol were found without any extra node.

Evaluation of diagram digitisation methods is further complicated as the ground truth information is often unavailable. This is a particular issue for the evaluation of text and connector digitisation methods. Manually labelling these components would require substantially more effort than symbol annotation. Therefore, the current evaluation of text and connector digitisation methods is generally subjective [44]. For instance, Mani et al. [44] used EAST [115] and Tesseract to digitise text in a set of industry P&IDs. They presented qualitative sample output detection and recognition results, however quantitative evaluation metrics were not used. Objective evaluation methods were used for text and connector digitisation in a limited number of cases. This occurred when ground truth data was available owing to the use of digital [65] or synthetic diagrams [43]. For example, Paliwal et al. [43] created a synthetic dataset of 500 P&IDs. The ground truth data of horizontal and vertical line locations, text locations and text strings were available. Their digitisation methods were evaluated on 100 synthetic P&IDs and a smaller private dataset of 12 real-world P&IDs. However, the text and lines methods were objectively evaluated on the synthetic dataset only. The text was considered correct if the string exactly matched the ground truth. Francois et al. [65] used text locations extracted from PDF engineering documents as the ground truth. A detection was considered correct if the predicted area corresponded to the ground truth area within an acceptable margin of 10 pixels.

The performance of text recognition methods can be objectively measured by comparing

the predicted string to the ground truth. This was seen in cases where digital or synthetic diagrams were used, or for a subset of the text. For instance, Nguyen et al. [63] extracted two specific text strings from technical diagrams. They applied the Exact Match accuracy for text recognition. The detected text string was considered to be correct if it exactly matched the ground truth. In another study, Kim et al. [32] used digital P&IDs for which the text ground truth metadata was available. In addition to text detection precision and recall, Kim et al. [32] also evaluated the combined text detection and recognition performance. More specifically, they used the Character Level Evaluation (CLEval) [141] metric to obtain precision and recall scores that combined text detection and recognition. CLEval [141] employs both instance matching and character scoring. Meanwhile, Khallouli et al. [134] evaluated their text recognition method using three metrics. These were character recognition rate, word recognition rate and average Levenshtein distance. The latter metric is the number of character edits (such as substitution, insertion or deletion) required to alter the predicted text to the ground truth text.

2.3.4 Class Imbalance

Class imbalance occurs when one or more classes are over-represented in a dataset. It is inherent in EDs as equipment types are represented with varying frequencies. The problem of class imbalance is known to occur in both deep learning and traditional machine learning [33]. Learning algorithms trained on imbalanced data are typically biased towards the majority class, which causes minority class instances to be classified as majority classes [142].

Class imbalance was shown to occur in both engineering symbols classification and detection [31, 1, 32, 60]. An example is the work presented by Elyan et al. [31], which showed that class imbalance effected the CNN classification performance of a P&ID symbols dataset. Lower performance on underrepresented classes compared to overrepresented classes was reported. In work on object detection, Elyan et al. [1] created a YOLOv3 [21] based method for symbol detection of an imbalanced dataset. Overall accuracy was high at 95%, although it varied across classes. A class accuracy of 98% for the majority class with 2810 instances was reported, whereas the accuracy for the minority classes with only 11 instances was 0%.

Similarly, Kim et al. [32] reported comparable results in their study on P&ID symbol detection. In particular, a lack of data for large symbols was reported. Lower class-accuracies were observed for underrepresented instances. Ziran and Marinai [60] also recorded imbalanced symbol distribution in two floor plan datasets. Interestingly, class representation was not strictly correlated with the performance of the Faster R-CNN

based model. The highest precision and recall values were not all for the most represented classes. This may be due to the high within-class diversity in the majority classes.

2.3.5 Contextualisation

In a previous review [13], authors defined *contextualisation* as the process of converting the digitised information (i.e. the shapes detected by the computer vision algorithms) into structured information, which can be used to better explore, manipulate or redraw the diagrams in more interactive and representative ways. In this subsection, we discuss the most common solutions in literature that have been presented for this purpose. We have split the contextualisation challenge into three sub-challenges: 1) *the storing challenge*, where systems have to be devised in order to save the structural representation in an easy to read/access manner, 2) *the connectivity challenge*, which refers to how the digitised objects are arranged in from their spatial representation in a way that users are able to know how symbols are connected and 3) *the matching challenge*, in which we address the issue of how to use these structural representations for real-life purposes, such as finding certain sections within a larger drawing, localising which portions of the drawing have relation to a 3D representation (i.e. the real facility or a digital twin), and ensuring consistency of the structural representation by inspecting it in semi-automated ways.

Since the earliest stages of P&ID digitisation, researchers have realised the need to convert the digitised information into some sort of structural graph representation to address the storing challenge. In the 90s, Howie et al. [143] proposed a symbolic model output with each of the shapes (symbols and pipes) as a node, and edges connecting them. This means that, despite pipes being connectors within the drawing, these should be represented as another node, as pipes themselves have their own attributes. A toy example is presented in Figure 2.8.

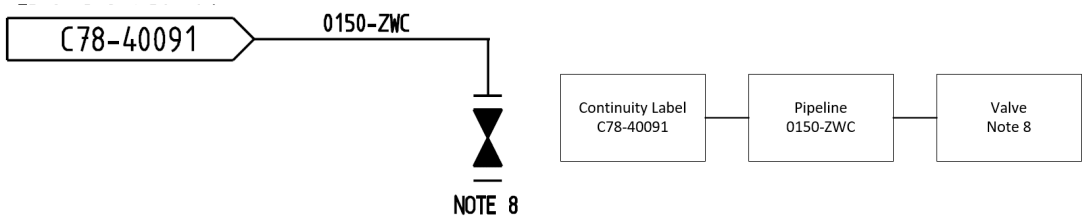


Figure 2.8: Left: A snippet of a P&ID with two shapes connected by a pipe. Right: The structural graph representation as proposed in Howie et al. [143]

To address both the connectivity and storing challenges simultaneously, other authors

have used the notions of graphs to find the connectivity between the symbols, bypassing the line detection. For instance, Mani et al. [44] used graph search to discover symbol to symbol connections in a P&ID. Each pixel was represented as a node, and links between neighbouring pixels were represented as graph edges. Then, symbol to symbol connections were determined using a depth-first search starting a symbol node. This approach would be of most benefit when drawings have a high quality and the algorithm can traverse from one symbol to another with relative ease. This system relies on connectors not overlapping with each other (since the graph search algorithm could be confused by the direction to take) and thus, have limited applicability when the drawing is complex and presents an entangled connector structure.

There are a handful of applications found in literature to address the matching challenge. For instance, Wen et al. [144, 145] presented a system to measure 2D-3D process plant model similarities based on their topological distribution, establishing a relation between a 2D ED and a 3D hydrocarbon plant model. To do this, each model was extracted as a graph, and then the feature similarity is calculated to measure a degree of matching between the two models using a geometric deformation invariant algorithm. Contrary to most of the literature reviewed in this study, authors used a type of CAD drawing called ISO drawing, which is relatively easier to digitise compared to classical EDs mentioned before (e.g. P&IDs) since it is more standardised and contains far more measurements and indicators. Still ISO drawings require vast knowledge and field experience to be correctly digitised and, therefore, the extraction of the attributed graph is done in a semi-automated way. Regarding the 3D plant, extracting the attributed graph is easier since the 3D model is still contained in a CAD file which retains all the meta-data needed for this reconstruction.

Rantaala et al. [146] also applied graph matching techniques to better use plant design information from older designs. Authors performed a review of graph matching techniques and evaluated six algorithms using an illustrative dataset built for purpose. In their evaluation, authors concluded that an algorithm based on simulated annealing with a certain combination of parameters was the best option for this task, as it was capable to detect spurious and inexact correlations. Later on, Sierla et al. [147, 126] presented related work on automatic generation of graphs from P&IDs. In this study the input was a P&ID represented in XML format, which was able to be converted into an attributed graph. To this end, authors used a recursive algorithm which also relies in pictures taken from the actual facilities, but that reconstruct the graph with an increased accuracy.

In more recent work presented by Rica et al. [148, 149], authors propose graph embeddings which are used to train neural networks on how to distinguish local substructures

which may be incorrect, this reducing the human effort on performing manual validation of the digitised information. To this end, authors first construct the graphs based on proximity information provided by the digitisation module, and then learn the most common substructures that can be found in the particular drawing set. For instance, a drawing may depict three valves connected in a loop, but no more than that. Afterwards, a GNN is trained to retain this information and validate the drawings. As in most graph-based problems, the complexity of this review increases with the size of the graph; therefore, authors tested this method in a smaller dataset.

2.4 Summary

Significant progress has taken place in the area of processing and analysing EDs and complex documents. This includes aspects such as symbol detection, text recognition, and contextualisation. A wide variety of deep learning models were used, for instance the literature shows that symbol digitisation methods are not only based on object detectors but also segmentation, classification and graph approaches. Meanwhile text digitisation methods were based on both specialised text methods and object detectors. Methods for connector detection have received comparatively less attention than symbol and text methods. Only 21% percent of the reviewed papers presented a method for connector detection. Overall, deep learning methods used for digitisation have proved to be beneficial compared to traditional methods and result in improved performance.

However, further research is still required to solve the timely and challenging problem of complex ED digitisation. Improved methods are still needed for the digitisation of all drawing components, namely symbols, text and connectors. This is challenging due to factors including diagram complexity, visually similar drawing components [42, 44], large intra-class variance [61] and low inter-class variance [43, 34], amongst others. The remaining key challenges for ED digitisation were identified as dataset acquisition, data annotation, imbalanced class distribution, evaluation methods and contextualisation. Although methods such as synthetic data generation and data augmentation exist, the literature suggests that further work is needed to address the specific challenges of ED digitisation.

This thesis advances the research in this domain as a deep learning framework for the automatic processing of engineering diagrams was presented. The framework automatically digitises text and symbols within these drawings. The text digitisation is presented in Chapter 3 and the symbol detection is presented in Chapter 4.

Another area that requires improvement is the data annotation process, which is typically time-consuming and consequently costly. A potential solution is to use deep

learning methods that learn from a few instances. This could be of benefit given the frequent presence of underrepresented and rare symbols within EDs. State-of-the-art methods such as few-shot learning are suggested. Unlike supervised learning models, which typically require vast amounts of labelled training data, few-shot methods aim to learn from only a few samples [150]. In this thesis, a few-shot symbol detection method is presented in Chapter 5.

Another critical need in this area is to develop and release datasets to the public domain in order to accelerate research and development. Real-world datasets are typically confidential however, datasets released publicly should ideally be of similar complexity and contain properties such as noise, overlapping elements and a wide range of symbols. This will ensure the findings are relevant to real-world scenarios. Furthermore, allowing researchers to use standard datasets would facilitate benchmarking of proposed methods. In this thesis, a symbol dataset is presented in Chapter 6. The dataset contains a large number of symbols from various classes and is imbalanced to reflect the symbol distribution seen in the real-world.

The literature also showed that the class imbalance problem occurs in engineering drawings. This is when deep learning models trained on imbalanced datasets become biased towards majority classes. This problem is addressed in Chapter 6, where a method is presented to improve multiclass classification of an imbalanced symbol dataset.

Chapter 3

Text Digitisation

This chapter aims to address the most fundamental challenge in ED digitisation, which is that of text digitisation. Here, text digitisation methods are evaluated for very complex real-world EDs. This involves two separate deep learning methods. The first method is designed for text detection and it aims to locate text instances within the drawing. The second method is designed for text recognition and it aims to recognise a text string from a section of the image predicted as text. This chapter also includes a thorough discussion of the findings to show where these methods perform well and which scenarios remain challenging. *This work was presented at the 2020 IEEE IJCNN [37].*

3.1 Introduction

It is common across many industry sectors for EDs to be stored in an undigitised file format or as a paper copy. Digitisation of these documents is of importance to allow improved use of this vast amount of data. EDs can be very complex and contain text annotations in addition to a number of different components including vessels, symbols and connecting lines. In digitising these documents, the detection and recognition of the text elements, also known as OCR, is a key part of the document digitisation. The ability to accurately read text in images is important for many applications. However, text digitisation in EDs is challenging due to factors including low resolution, image noise and overlapping elements.

Prior to the use of deep learning in text detection and recognition, traditional methods were used. This involved extracting features, which were predominantly low level or mid level, in a process which required many pre and post processing steps. Colour, texture and edge features were often used for text localisation. Approaches commonly used

connected component analysis or sliding windows [113] for that matter. In particular, a family of approaches known as *Text/Graphics Separation* (TGS) methods [151] were used for drawings such as general purpose EDs [152], circuit diagrams, maps [153] and musical scores, with moderate success.

Deep learning methods have the potential to improve detection systems in computer vision, remote sensing and cybersecurity amongst other domains. Examples include deep learning methods for the recognition of targets in Synthetic Aperture Radar images as in [154]. Authors presented a method using a CNN trained with a cost function to which intra-class compactness and inter-class separability information was added. A SVM classifier was used and results showed an average recognition accuracy of 99% across ten target types. In the field of cybersecurity, an innovative intrusion detection system based on a statistically driven deep autoencoder was proposed in [155]. Evaluation on binary and multi-classification tasks showed the suggested autoencoder method, designed with a single hidden layer of 50 units, achieved higher performance in comparison to other deep and traditional algorithms.

In recent advancements related to computer vision, deep learning has greatly improved object detection methods. Whilst text has specific properties in comparison to objects in the generic object detection task, object detection models can also be used for text detection. For example, in [156] authors adjusted the popular YOLOv3 model [21] in order to detect text.

Whilst text detection may be viewed as a specific type of object detection, text detection has specific challenges in comparison to general object detection, and thus it is relevant to create specific methods tailored for this task. A range of deep learning methods were developed for text detection, including the EAST [115], CTPN [116], TextBoxes [157], TextBoxes++ [158] and Fastext [159].

3.2 Common Existing Text Digitisation Methods

Text detection and recognition methods can operate at two levels: character level and word level [160, 112]. Character level methods rely mostly on heuristic-based segmentation techniques that are able to distinguish text from other shapes based on innate characteristics of the letters and/or numbers such as size, stroke and geometry. After the identification of letters occurs, these methods rely on techniques to constitute strings (mostly through character proximity and alignment) and to classify each character individually to interpret such strings. In contrast, word level recognition is primarily suited to situations where there is a restricted number of possible words in a document, allowing lexicons to be used in conjunction with character recognition outputs. This

may only be applicable to EDs when there is a set of standardised codes that have a fixed structure. For instance, a lexicon of words to narrow down the number of possibilities for word recognition was a strategy used to improve the OCR accuracy in [161].

As stated in Section 3.1, TGS methods were initially a very popular option for text detection in the document image analysis community some years ago, given their simplicity and robustness [13]. One of the cornerstones of this area was the work presented by Fletcher and Kasturi in 1988 [162], where authors proposed the use of CCA to select text characters based on a predetermined size and width-to-height ratio. Afterwards, the resulting text layer was converted into its Hough transform to analyse the linearity of the characters and deduce the strings conformed. This approach was very favourable for simple EDs, however it was incapable of dealing with text overlapping shapes and with short strings (i.e. less than 3 characters of length). A number of reviews and upgrades were done to this working pipeline, mostly at the string conformation stage, such as Lu [163] who used a *brushing* morphological operation to join strings, Tombre et al. [151] who were able to discard dashed connectors from the text layer and applied proximity analysis for the string generation, and Tan and Ng [164] who by using a *pyramid* approach were able to scale down the text layer until being able to find the optimal string conformation. Most recently in [165], authors presented a comparison of different TGS methods to reduce the overall identification of shapes and connectors in P&IDs. It is worth to note that for the text recognition stage, most state-of-the-art methods rely on OCR for the text interpretation, nonetheless there is work in literature where character classification is preferred as it is more suited for EDs. A study by Das et al. [166], involved identifying areas of text in architecture, engineering and construction documents through traditional methods, however the study did not attempt to read the text instead focussing on classifying text as either machine printed or handwritten.

One essential drawback of TGS methods is their general inability to deal with text overlapping other shapes of the ED. Although some work has been presented in this matter [167], this usually relies on a series of heuristics that are not always applicable and thus have various rates of success depending on the overall quality of the ED. Moreover, it has been noted by authors such as Ye and Doermann [112] that general object detection methods would not perform well for text detection in a more general setting, based on a comparison of average images of three object types namely faces, pedestrians and text. In their experimental setting, the average images were composed of the mean of $2,000$ aligned samples of each object type and whilst the face and pedestrian image retained common features, the average text image resembled noise.

In methods more related to the domain of EDs, and specifically P&IDs, Sinha et al. [54]

presented work on extracting text information from scanned raster versions of P&IDs. The proposed method however focussed only on text within tables, and used initial steps including contour detection to detect tables in the diagram. The method was tailored specifically for P&ID dataset used, with the tables detected having to match one of three specified formats containing specific keywords. To extract the table information, version 3.05 of Tesseract OCR and Python RegEx string matching were used. The method correctly identified 87.2% of the tables present, however inconsistencies in the information extraction occurred, potentially due to some text touching table borders and logos appearing as text.

In [168], a study on detecting characters in EDs was presented that used a convolutional object detector based on Overfeat [169], Faster R-CNN [88] and Feature Pyramid Networks (FPN) [170]. The detector took a single image as input without pre-processing and output class confidence scores and bounding box predictions. The system was tested on a dataset of 150 EDs and results only showed passable accuracy with some misclassifications and false negatives. Moreover, Eman et al. [171] presented work aimed at improving OCR accuracy in complex cursive scripts, using conditional GAN to transform cursive text into straight scripts, where characters are not joined, before LSTM based OCR was carried out. Results, evaluated on character level error rate with the Levenshtein distance, showed improvement with the recognition of handwritten and italicised cursive scripts.

Traditional methods for text detection were compared with deep learning methods on text in floor plan images in [161]. The analysis compared four methods: 1) EAST, 2) CTPN, 3) a standard image processing approach using MSER, SWT and Tesseract to discard areas of non-readable text, and 4) a combined approach with all of the first three methods. For the CTPN method, additional sub images along the border were used as CTPN struggled with identifying text close to the image borders [161]. The combined method compared results from all three other methods against each other to produce an output based on voting. Post-processing was carried out on all methods to merge specific text boxes into one text item. The text was firstly classified based on rules, then room descriptions were compared with a dictionary of valid words and replaced with the closest word based on edit distance and word frequency. The proposed methods were evaluated on datasets of varying quality. Performance with the CTPN method was shown to be significantly reduced by the noise and low resolution images. On the low quality images, the EAST method had the highest recall and F1-score, whilst the combined method had the highest precision. None of the proposed methods were able to detect vertical or curved text items and the accuracy of the recognised text wasn't analysed in detail, however it was noted that Tesseract did not give correct predictions

on the low resolution images.

All of the aforementioned methods work with a varying degree of success for complex EDs such as P&IDs, as this type of printed drawings present multiple challenges, such as a dense and entangled structure between shapes, a complex hierarchical relation between elements, overlapping of text with other shapes and the similarity between symbols and text, amongst others.

3.3 Methods

3.3.1 Dataset

EDs used in industry are not widely available in the public domain primarily due to data confidentiality reasons. To evaluate the methods on real-world data we have, through collaboration with an industry partner, obtained a dataset of P&IDs. Note that this is one of several datasets used in this thesis. The dataset we have chosen to use will allow the selected deep learning methods to be evaluated on real world complex engineering diagrams; the P&IDs in the dataset are from the oil and gas industry however P&IDs are also used in many other industry sectors to convey and store information about process equipment and its operation.

The dataset comprises 172 complex P&IDs, which contain components of symbols, connector lines and text annotations. A section of a P&ID, rather than the whole diagram for data confidentiality reasons, is shown in Figure 3.1. This figure represents part of a typical P&ID containing symbols for different valve types, text describing the valves and tag numbers, and equipment connectivity information in the form of pipelines and electrical connections.

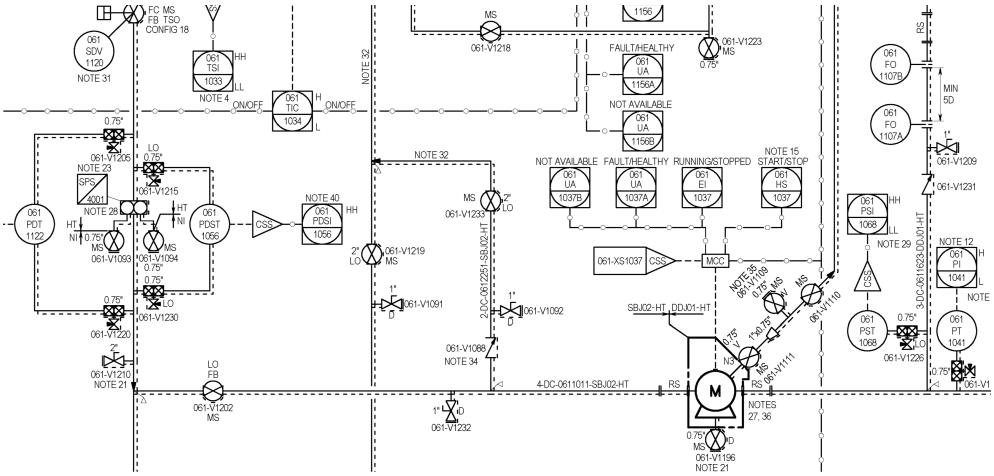


Figure 3.1: Section of P&ID showing symbols, connectors and text elements

Text in the P&IDs dataset analysed, can be split into two types according to its purpose and location. Text located in the main diagram is used to annotate the graphical elements, including equipment tags whereas the second type of text is located within the diagram template and is used to provide additional details including drawing number, revision history, and further related details about the equipment shown in the diagram. Although the ground truth text data was unavailable, analysis of a subset of P&IDs showed that in the main diagram section, there are approximately 415 text instances per P&ID.

Text within the P&IDs is used to annotate the equipment in the diagram and includes text that ranges from short text strings including two character annotations, through to line numbers and equipment tags, to longer full sentences showing operating information for equipment in the diagram. Text is located vertically in addition to horizontally, one such situation where vertical annotations are used occurs where an associated line number is aligned next to a vertical pipeline. There are also several text strings printed diagonally to align with equipment. Text is located throughout the diagram, with some text in close proximity to other components and there are text annotations situated within symbols and vessels. In P&IDs parts of some non text components are similar in appearance to text characters and certain elements such as dashes occur in both text and non text components. Dashes are present in a large amount of text strings such as equipment tags, whilst dashed lines are used as a form of connection line between two pieces of equipment and located adjacent to pipelines to indicate a property of the line.

Therefore the images in the dataset chosen, contain several challenges related to both text detection and text recognition and are suitable for evaluation of deep learning methods applied to digitise text from within complex EDs.

A schematic diagram depicting the proposed steps in text digitization from the complex P&IDs is presented in Figure 3.2.

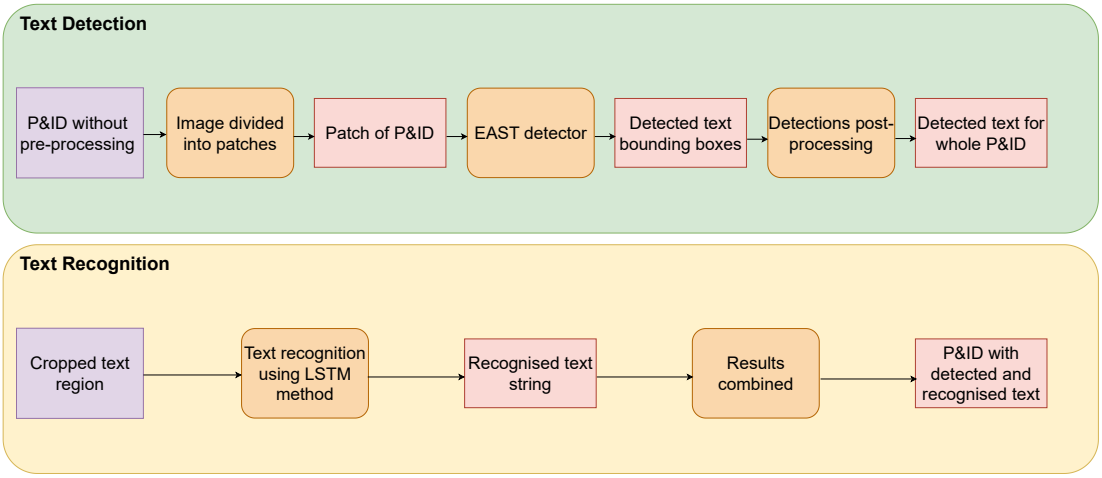


Figure 3.2: Schematic diagram of P&ID text detection and recognition. Inputs are shown in purple, processes are shown in orange and outputs are shown in pink.

3.3.2 Text Detection

A deep learning method, the EAST detector [115], was chosen to detect text instances in the P&IDs. The EAST detector is reported to outperform other state of the art methods when evaluated on F-score on the text detection tasks COCO-Text [130], ICDAR 2015 Challenge Text Localization Task [172] and the MSRA-TD500 [173] dataset.

At the time of EAST proposal, existing methods were commonly designed with several stages [174, 175]. The EAST detector does not contain any intermediate steps like candidate proposal, instead it produces text predictions directly from a single neural network. Output results from the neural network are then filtered using a very similar process to Non Maximum Suppression (NMS) where the geometries are averaged as opposed to being selected.

More specifically, EAST uses a FCN which is trained to predict word or text line instances from full images. The network was based on the general design of DenseBox [176]. The FCN architecture used in EAST can be split into a feature extractor stem, feature merging branch and an output layer. The purpose of the feature merging branch is to improve detection of both small and large word regions by merging features from both lower and higher layers of the feature extractor. The output channels consist of a score map in the range [0,1] which represent the confidence of the geometry shapes, which are predicted in the other output channels. Output geometries are predicted as a rotated box or quadrangle.

The loss function used in training the model comprises the loss for the score map and

the loss for the geometry predictions. Zhou et al. [115] selected a balanced cross-entropy loss for the score map. A scale invariant loss is selected for the geometry, to ensure accurate predictions for both large and small text regions. The two loss functions chosen for the geometries are IOU loss for rotated rectangle output predictions, and scale normalised smoothed L1 loss for the quadrangle geometries. Zhou et al. [115] trained the EAST network end-to-end using the ADAM optimiser [177].

3.3.3 Text Recognition

A deep learning method, specifically LSTM networks [178], was used for text recognition. This network is a type of RNN designed to retain information from long sequences. This was implemented using the open source OCR engine Tesseract v4 [119], which uses an LSTM neural network to recognise text strings from text lines.

3.3.4 Pre-Processing and Post-Processing Data

Selection of Input Image into EAST Detection Model

In the dataset used, the ED had a template format with a table that was consistently located on the right hand side. The focus in this study was to interpret the main part of the diagram itself, therefore the text detection and text recognition will be applied only to text located in the main diagram and not the text information in the diagram template.

The method selected to discard the template was based on connected components. It was chosen as it is independent of template layout, as the method does not require heuristics based on the position of the template within the diagram. The method used to select the diagram area would also be applicable to other datasets, including those without a border line.

The largest white component by area represents the background of the main diagram itself, and thus the connected components algorithm method was used to determine the largest connected component in the P&ID by area in order to select the diagram area to be processed by the text detection model.

The P&ID diagrams were large images, approximately 7500 x 5250 pixels in size. To process the whole diagram area in one step by the EAST model would need a high amount of computational requirements therefore to reduce processing requirement, the diagram is processed by the EAST detector in four patches. The image patches to be processed were obtained by dividing both the height and width of the selected diagram area in half.

Post-Processing of Text Bounding Boxes

Padding was applied to the detected text boxes, using image pixels, to ensure that all of the text string was included in the bounding box. To make the method applicable to text regardless of font size, the amount of padding added was calculated as a percentage of the original detected text box size. The height of the text box was padded by 10% and the width was padded by 12% at the start and 24% at the end of the string. The variables used in the padding step were found experimentally to obtain the optimum text recognition results.

A post processing step was then taken to merge nearby detected text boxes based on the proximity of the bounding boxes. Detected text boxes were split into horizontal text or vertical text based on the ratio of the width to height of the bounding box. The area of overlap between each pair of detected text boxes was calculated and if text boxes overlapped, they were combined into the smallest bounding box that would combine both original detected boxes. The resultant bounding boxes were then used as input for the text recognition step.

3.4 Experiments

3.4.1 Setup

Experiments were run to evaluate the performance of the selected deep learning text detection and text recognition methods on complex EDs. To evaluate the chosen methods on real world data, experiments were performed on the dataset of 172 P&IDs from the oil and gas industry.

One challenge in using deep learning models for text detection and recognition in P&IDs is that there is no publicly available dataset of P&IDs annotated with text ground truth location and text string. This experiment identifies scenarios where pre-trained state-of-the-art deep learning models for text detection and recognition perform well and scenarios where improvement is required. Therefore the models were not trained specifically using the text from the P&IDs, and the whole dataset of 172 P&IDs is used as the testing set.

The text detection and text recognition methods were applied on the P&IDs by creating a framework to process the diagrams. Results were evaluated by displaying the results on the processed P&ID. Bounding boxes were shown on the detected text instances, with the output string from the text recognition step shown adjacent to the detected bounding box. Additionally for evaluation purposes, output files listing the detected bounding box co-ordinates, dimensions and predicted text output, were also produced.

A pretrained EAST model [115] was used to locate text instances. One of the challenges in analysing the P&IDs with deep learning methods is the relatively large image size, on average 7500 x 5250 pixels, which consequently has high processing requirements. The P&IDs were therefore processed using patch detection. The patch detection method works by splitting a larger image into smaller patches which are processed in turn by the deep learning method, with the output detections from each patch combined to obtain the detections relative to the whole image. Text strings located across more than one patch were split into multiple sections when input to the detection model.

To perform text recognition, open source LSTM based Tesseract engine was utilised. Speed of processing the diagrams was an important factor in this study, therefore the smallest LSTM network from Tesseract was chosen as this had the fastest processing speed available, however this LSTM model was also associated with decreased accuracy levels compared to the larger LSTM network model available in Tesseract.

3.4.2 Results & Discussion

The P&ID images produced from the experiments with results overlaid show that the EAST detection model and LSTM based text recognition method gives promising results when applied to detect and read text in complex engineering diagrams. Results from experiments on the real world P&ID dataset are discussed in further detail below.

In experiments, the EAST model was able to detect varying orientations of text, as stated in [115], with both horizontal and vertical text instances in the diagram being detected. The model detected text strings of varying lengths. A sample of text instances that were correctly detected and recognised have been extracted from the analysed P&IDs and are shown in Figure 3.3. The bounding boxes indicate the areas in the image detected as text and the text string predicted by the text recognition model is shown adjacent to the detected text area.

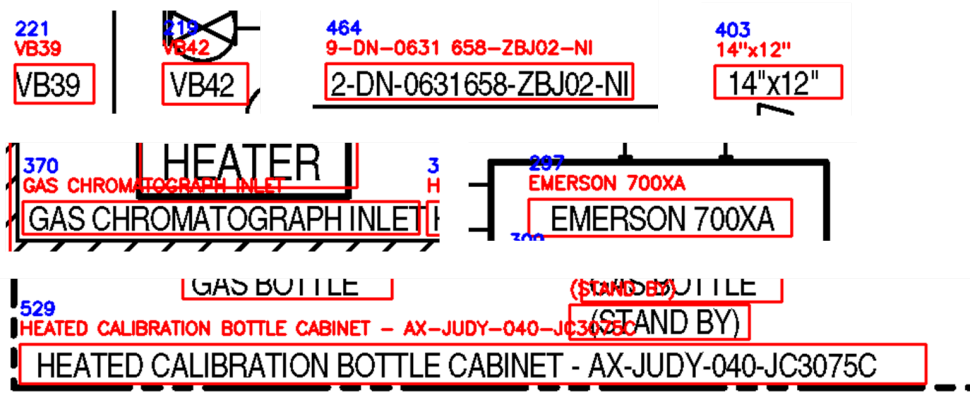


Figure 3.3: Instances where text was correctly detected and recognised. The detected text boxes and recognised text strings are shown in red and the output item numbers are shown in blue.

To analyse the results in further detail, from the dataset of 172 P&IDs we have selected eight representative P&IDs for which to present the quantitative text detection and recognition results. Note that a subset of drawings was used here due to the manual effort required to obtain the ground truth. Table 3.1 shows the numbers of text instances present in each diagram, the percentage of detected instances, FN and FP detections. The percentage of text detections with associated text string from the recognition step is also listed. Additionally, the distribution of the result variables is shown in Figure 3.4.

Table 3.1: Analysis of text detection and recognition on selected P&IDs

Diagram No.	Text Instances	Detected (%)	FN (%)	FP (%)	Recognised (%)
1	426	87	13	4	87
2	492	91	9	3	83
3	545	89	11	4	87
4	407	91	9	4	86
5	201	92	8	5	86
6	544	84	16	3	87
7	276	91	9	4	86
8	427	93	7	5	82
Mean	415	90	10	4	86
Std. Deviation	122	3	3	1	2

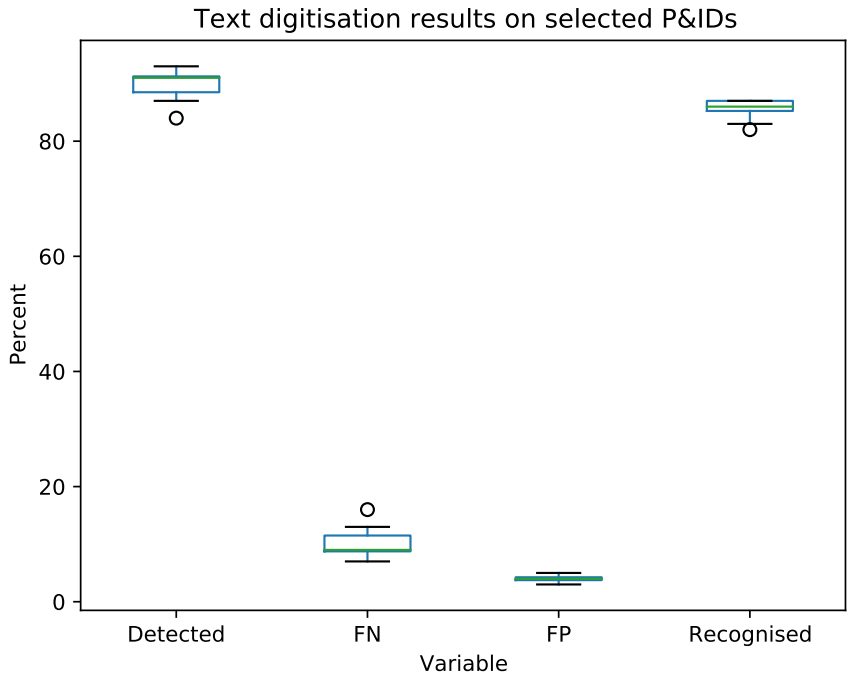


Figure 3.4: Text digitisation results on selected P&IDs.

As shown in Table 3.1, there are on average 415 text instances, counted as one text string or multiple text strings that would be combined into one detection by the post-processing, present in each diagram. The standard deviation of number of text instances per drawing was 122, which indicates that this variable can vary substantially between drawings. When images were passed to the EAST detection network, 90% of the text instances were successfully detected, without the need for any pre-processing of the image or training on the specific font from the P&IDs.

One challenge particularly related to text detection in P&IDs is the presence of other diagram elements, or sections of, that resemble text characters. False positive detections, where non-text elements of the diagram were detected as text and the detected areas contained no text characters, were observed to occur on average in 4% of output detections based on the sample set analysed. A sample of false positive detections has been extracted from the processed P&IDs and is shown in Figure 3.5 to highlight examples of P&ID sections where the model does not accurately distinguish between certain drawing components and text characters. False positive detections contain P&IDs diagram elements that resemble text characters including dashed lines, parts of valve equipment symbols and triangle pipeline flow indicators.

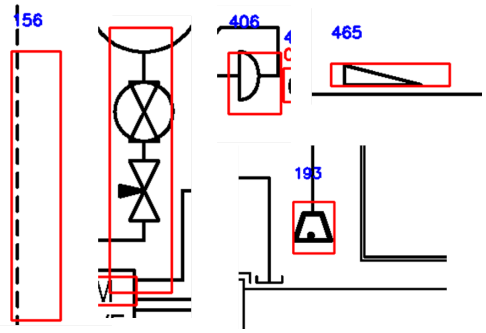


Figure 3.5: Instances of P&ID elements misdetected as text. The detected text boxes are shown in red and the output item numbers are shown in blue.

There were also instances of text that were not detected by the chosen method. Results show that on average, approximately 10% of text instances were undetected in each P&ID.

The standard deviation of detected, FN and recognised text instances was 3, as shown in Table 3.1. That of the FP detections was less at 1. Additionally, the distribution of the result variables can be seen in Figure 3.4. These results indicate that the deep learning text digitisation models performed relatively consistently across the drawings in the sample dataset.

Text Detection

In certain scenarios it was shown to be a challenge for the deep learning detection method to accurately distinguish between text and non text elements. Incorrect bounding boxes round the text strings were observed to occur in three scenarios, 1) partially detected text string, where one or more characters is determined not to be text by the deep learning model, 2) non-text elements determined to be text data and 3) non-text components included in the text area bounding box as a result of the post processing steps of merging text boxes and padding. Furthermore, it is possible for combinations of these scenarios to occur in one output text box.

There are many technical annotations used in the engineering diagrams, including line numbers that often start with a single character followed by a dash, in many instances the start of this text string was missed from the detected bounding box.

It was also observed that when text was located in close proximity to other components, the non text components could be included in the bounding box, likely due to the padding applied in the post-processing.

One of the images in the dataset consisted of a table containing line numbers, rather

than a diagram. Text was detected in every cell of the table. The only text strings not detected were short strings of two letters. Additionally some of the text was joined into blocks and detected, and in several instances the string was partially detected.

Text Recognition

Results of the detection step feed directly to the recognition step, therefore obtaining a good output from the detection step allows a cleaner image of the text string to be passed to the text recognition step.

Whilst the EAST model was able to detect text in the vertical direction, instances where the vertical text appeared to have been read in the wrong direction were observed, refer to Figure 3.6.

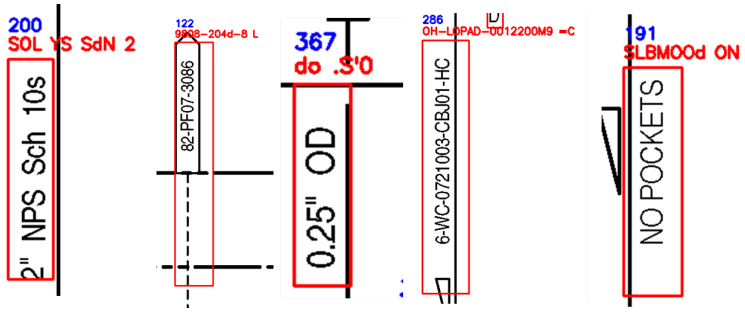


Figure 3.6: Incorrect recognition of vertical text instance. The detected text boxes and recognised text strings are shown in red and the output item numbers are shown in blue.

3.5 Summary

Deep learning methods have brought advancements in the area of image text detection and recognition. However, there was a lack of in-depth analysis of deep learning models applied to the problem of digitising text information from complex EDs.

In this chapter, state-of-the-art deep learning methods were used for text digitisation from engineering diagrams, namely the EAST model [115] for text detection and an LSTM method for text recognition. Evaluation of these methods was carried out on a dataset of 172 complex P&IDs from the oil and gas industry. Experiments showed that without pretraining the deep learning models with the P&IDs text data, the models correctly detected and recognised certain text strings that occurred in simpler scenarios. Text instances that were located in more complex scenarios proved to be a challenge for the methods used.

A suggestion for future work is to aim to increase the deep learning model accuracy by

pretraining the models on the text used in the P&IDs. A specific focus of further work will be on training the deep learning models for accurate detection and recognition of text strings that are located in close proximity to non-text diagram elements.

In addition to the text, one of the other main components used in complex EDs is symbols. These represent a wide variety of equipment and thus their information is crucial to understand the system depicted in the drawing. In the following chapter, deep learning research is presented for the difficult problem of symbol digitisation within these drawings.

Chapter 4

Symbol Detection

This chapter presents a novel framework for the automatic processing and analysis of construction drawings. These drawings have received considerably less interest compared to other drawing types. However, analysing them typically requires substantial manual effort for many crucial tasks, such as material takeoff where the purpose is to obtain a list of the required equipment and respective amounts for a project. This chapter presents experiments using two different approaches for symbol detection in challenging high-resolution drawings sourced from industry. The framework presented allows for the digital transformation of construction drawings, improving tasks such as material takeoff and many others. *This work has been published in the IJDAR [38].*

4.1 Introduction

Construction drawings are essential documents as they show what will be built for a project. These drawings are still frequently stored in undigitised formats, and consequently, retrieving information from them must be carried out manually. This requires domain experts [5], and can be very time-consuming [4] and costly.

One of the most important processes in a construction project is material takeoff or quantity takeoff [179, 5]. The purpose of this task is to create a list of the required materials and quantities. The list is an essential document as it is used for cost estimation [179]. It is important that it is accurate, as any errors can impact the project budget and schedule [7, 180]. The takeoff is traditionally carried out through manual drawing analysis. However, this can be time-consuming and prone to counting errors, particularly for large projects [4]. Furthermore, the results are dependent on individual interpretations [7].

Artificial Intelligence (AI) based methods can augment employees' capabilities by simplifying time-intensive and repetitive tasks [2]. The use of state-of-the-art digital technologies to transform traditional industry practices into autonomous systems has been referred to as Industry 4.0, or the fourth industrial revolution [181]. The number of publications that discussed AI applications in the construction industry has increased in recent years [182]. Within this, one of the main research topics was computer vision [182], where the applications mainly focussed on the monitoring of construction sites and structural health. However, the current adoption of AI-based applications in the building and construction industry is relatively low [181], and lags behind that in other industries [182, 183].

Across a range of sectors, there has recently been an increasing demand to create methods to digitise EDs [44, 45]. This involves extracting all diagram components, which are the symbols, text and lines. Although published research on this topic dates back to the 1980s [12, 9], as stated in a recent review by Moreno-Garcia et al. [13], digitising complex EDs is still considered challenging. For instance, deep learning methods were used for symbol digitisation in other types of EDs [1, 3]. This was considered a difficult task for multiple reasons, including the numerous symbols present in each diagram [1], relatively small symbol sizes [1, 3] and use of non-standard symbols [32, 184].

Although construction drawings are more complex compared to other types of EDs, methods for their digitisation have received considerably less attention than those for other ED types, such as P&IDs [1, 32, 3, 44, 56, 8, 51]. One reason that construction drawings are more complex is that they are typically composed of multiple drawing layers. These organise graphical elements by type [185], and can be shown overlapping each other. This means that symbols are typically shown on a highly complex background. Additionally, these drawings contain a significant amount of visually similar shapes. Furthermore, they are typically grayscale and thus, no colour information is available to help distinguish between components.

This chapter presents a novel deep learning framework to process construction drawings automatically. It should be noted that EDs are generally unavailable in the public domain [45, 13] due to confidentiality reasons. Therefore, in this experiment, a dataset was obtained from an industry partner to ensure that the research is relevant to a real-world scenario. Multiple building systems are represented, including plumbing and HVAC. The drawings are very complex and contain various symbol classes, typically shown on a cluttered background, as shown in Figure 4.1.

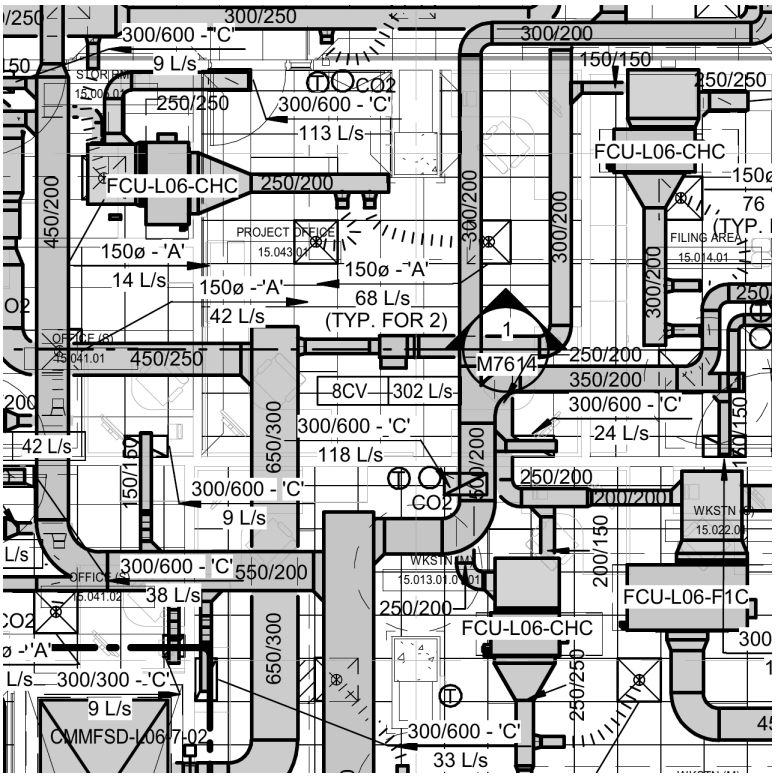


Figure 4.1: Section of an 'HVAC' drawing. This is challenging to digitise for multiple reasons, including the dense representation of equipment, overlapping components, and complex background

4.2 Symbol Recognition Methods

Deep learning methods for ED digitisation were only proposed very recently. Symbol digitisation methods were mostly based on object detection models [36], which predict the class and bounding box of target objects in an image. Most research focussed on one type of object detector, such as YOLO [19] based approaches [52, 72, 11, 58, 45, 1, 75] or Faster R-CNN [88] based approaches [2, 63, 57, 4, 71]. Other approaches were based on FCN [90] segmentation models [34, 43] or graph-based methods [92, 6, 62].

The literature on symbol digitisation in EDs covers a range of drawing types, with a particular focus on P&IDs [1, 32, 3, 44, 56, 8, 51]. For instance, Elyan et al. [1] created a YOLO-based method to detect symbols in P&IDs. They reported high performance overall with an accuracy of 95%, although the results varied across the symbol classes. Meanwhile, Gao et al. [3] presented a Faster R-CNN-based symbol detection method. On a dataset of publicly available nuclear power plant drawings, they reported mAP values of 92% and above for three separate groups of symbols. In another example on P&IDs, Mani et al. [44] created a CNN-based classification method for fixed size drawing patches. They obtained promising results for two symbol classes, however this

method may be computationally slow when scaled up for a larger number of classes.

There is also published research on symbol digitisation in architectural floor plans [186, 2]. For instance, Rezvanifar et al. [186] presented a YOLO-based method for symbol detection. They evaluated the method on a private dataset aswell as the public SESYD dataset. On the latter, they showed that their method outperformed traditional symbol spotting approaches. Note that this dataset contained drawings that are simplified compared to real-world drawings, for example they contained no occlusion, clutter or interclass variability [61]. In the same domain, Jakubik et al. [2] presented a human-in-the-loop approach for the detection and classification of symbols. They used a Faster R-CNN-based symbol detection method that was trained using a synthetic dataset created using a data augmentation approach.

However, there was a lack of research on construction drawings, as there are only a few recent works that presented deep learning methods for generating a list of materials from construction drawings [4, 180]. Joy and Mounsef [4] presented a Faster R-CNN based method to automate material takeoff from electrical engineering plans. They used a dataset of five drawings. Training data was generated using symbols extracted from the legend and image processing-based data augmentation. Whilst the method did not require extensive manual annotation, it relied on a suitable legend being available. Prior to testing, the background and text strings were removed. An automated method based on Tesseract was used to remove the text. This may be particularly important here, as these components were not included in the testing data.

Chowdhury and Moon [180] presented a Mask R-CNN [91] based method to automatically generate the bill of materials (BOM), which is a list of the required item quantities and costs, from 2D images of concrete formwork. Mask R-CNN is an object segmentation model, which predicts pixel-level object masks rather than bounding boxes. They created 206 drawings from 3D models, which were relatively clean with few components. On the validation data, a mAP of 98% was reported. The method showed promising results on the test drawings, however detailed metrics were not presented. On an actual construction shop drawing, the increased complexity meant that pre-processing was required to remove unnecessary elements. The solution relied on the manual selection of relevant items within a cost database to produce the BOM.

In a related area, published work discussed automated quantity takeoff from Building Information Modelling (BIM) models [187, 5]. BIM has played the leading role in digitising the construction industry [182], and it concerns the creation of a 3D model to manage building data. The drawback of BIM-based takeoff approaches is that considerable time and resources are needed to create the BIM. Furthermore, errors in the

BIM impact the accuracy of the quantity takeoff [187].

The literature shows that although deep learning has significantly improved computer vision methods, there is a lack of progress in construction drawing digitisation methods. Deep learning methods for ED digitisation were proposed only very recently, and these were primarily focussed on other ED types [1, 32, 3, 44, 56, 8, 51]. Moreover, there was a lack of research showing how different deep learning object detection models performed on complex real-world construction drawings.

4.3 Methods

This section discusses the methods used in the proposed symbol detection framework for complex construction drawings. This includes a discussion of the real-world dataset used for evaluation purposes.

4.3.1 Dataset

Overview

A dataset of 198 PDF construction drawings was obtained from an industrial partner. It contains three unequally represented types, as there are 92 'plumbing', 103 'HVAC' and 3 'other' drawings. To prepare the dataset for the experiment, the PDFs were converted to high-resolution 14,042 x 9,934 pixel PNG images at 300 dots per inch (dpi), as shown in Figure 4.2.

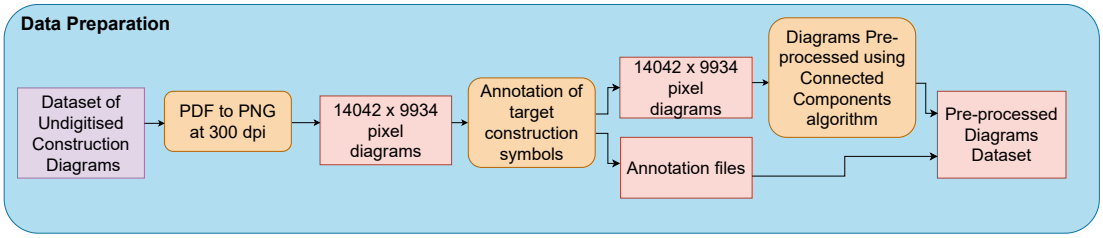


Figure 4.2: Data preparation steps. Inputs are shown in purple, processes are shown in orange and outputs are shown in pink.

The diagrams contain numerous symbol classes, 13 of which were selected for the experiment. These were chosen as they are required in the takeoff, and are shown in multiple building systems. The 'Detail Legend' and 'Direction of Flow' symbols were also included, as detecting them can help to determine links between diagrams or flow direction [29].

The symbols of interest are represented by various shapes, as shown in Figure 4.3. It should be noted that these examples were cropped from the legend, and are thus

displayed on a white background, unlike typically seen within a diagram. The symbols are challenging for a model to detect for several reasons. Each symbol is only represented by a few lines or shapes and thus there are only a few features available for a model to learn from. They are commonly represented in different orientations, with different shading and often overlap other shapes. Intra-class variability in the graphical notations was also seen, as shown in Figure 4.4. Furthermore, there is high inter-class similarity, for instance, the shape that represents a Gate Valve is also part of the Automatic Control Valve (ACV) and the Valve and Capped Provision (V&C Provision).

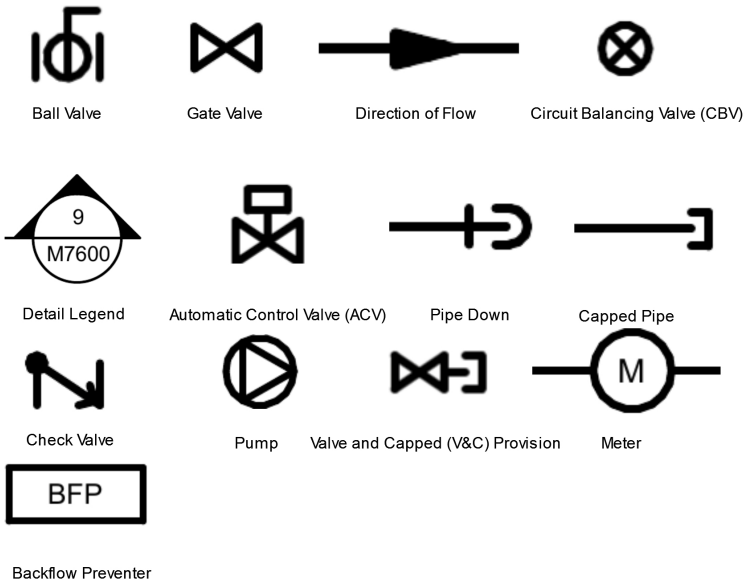


Figure 4.3: Symbol legend

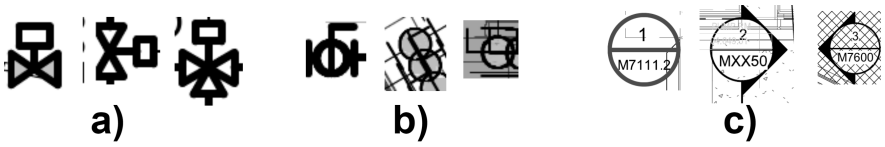


Figure 4.4: Examples of intra-class variability. The symbols in each group represent the same class, which are a) ACVs, b) Ball Valves and c) Detail Legends.

Data Annotation

The diagrams were manually annotated to create a symbol dataset, which can be a very time-consuming and demanding task [72, 29, 188]. The process includes drawing bounding boxes closely around each target symbol. For the purpose of the experiment, the diagrams were manually annotated using Sloth [1]. This is an open source tool which

[1] https://sloth.readthedocs.io/en/latest/

allows for object annotation. It should be noted that the annotations are exported to one output file per diagram. These record the labelled symbol information, including the class and bounding box coordinates. In total, the symbol dataset consists of 6,231 symbols from 13 classes. The manual annotation process took 90 hours to complete.

4.3.2 Data Exploration and Pre-Processing

Different equipment items are shown with various frequencies within construction diagrams, therefore the symbol dataset is highly imbalanced, as shown in Figure 4.5. Class imbalance is a major problem in both machine and deep learning [33, 142, 189] and is when algorithms trained on an imbalanced dataset are biased towards the majority class. It was observed that the Ball Valve symbol is significantly overrepresented, as it constitutes 35.3% of the dataset. In contrast, the four least represented classes each constitute less than 1%.

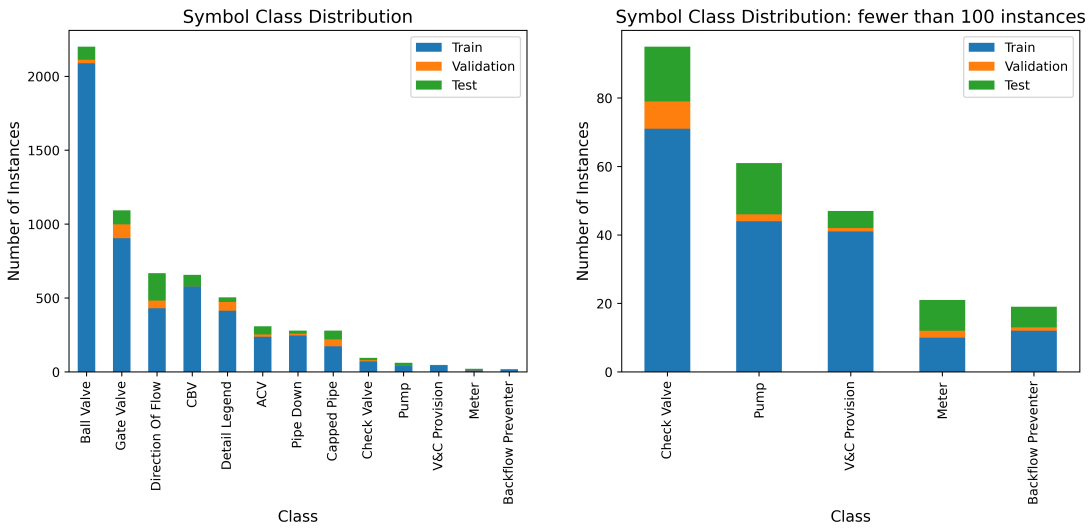


Figure 4.5: The left image shows the class distribution across the whole symbol dataset. The right image shows the distribution amongst those classes with fewer than 100 instances in more detail

The problem of small object detection was also seen in this experiment. Most of the symbols are smaller than 100 x 100 pixels. This is considered a problem due to reasons such as limited context information and indistinguishable features [190, 191]. For example, on the COCO dataset [191], the AP of YOLOv7 for small objects was lower, 35.2%, compared to that for medium objects, 56.0%, and large objects, 66.7% [25]. It should be noted that in the COCO dataset, the objects were classed as small if their area was less than 32 x 32 pixels, medium if between 32 x 32 pixels and 96 x 96 pixels, and large if more than 96 x 96 pixels [192].

The diagrams were pre-processed to reduce false positives, as shown in Figure 4.2. This was done by removing the diagram border, which contains text and no target symbols. A Connected Component (CC) algorithm was used to locate the largest CC of white pixels, which was considered to be the background of the main diagram area. In this calculation, the pixels were defined as connected to each other if they had four-way connectivity. An image mask was then applied to replace the pixels outwith the bounding box of the largest CC with white pixels.

4.3.3 Symbol Detection

Two state-of-the-art object detection models were used in the experiment. These are YOLOv7 [25] and Faster R-CNN [88]. YOLOv7 [25] is a variant from the YOLO series [19, 20, 21, 22, 23, 24, 25, 26]. It is a one-stage model, that predicts objects' locations and classes using a single CNN. It is known for its fast performance, for instance YOLOv7 had improved speed and detection accuracy on the COCO dataset [191] compared with other object detectors in the range 5 to 160 fps [25]. Additionally, YOLO also performs well across different types of diagrams [1]. Faster R-CNN [88] is known to be accurate, with state-of-the-art performance on the PASCAL VOC benchmarks [193]. Whilst Faster R-CNN [88] improved on the speed of earlier related models, Fast R-CNN [88] and R-CNN [80], its separate region proposal stage results in slower speeds compared to one-stage models.

The construction diagrams are significantly larger compared with the typical image input size for deep learning object detection models. For example, the diagrams are $14,042$ x $9,934$ pixels whereas the YOLOv7 input size is 640 x 640 pixels [25]. Using the whole diagrams as training images would require considerable computing resources and therefore, a patch-based approach was used. This involves splitting high-resolution images into smaller patches [194, 61, 1]. In this experiment, the patch size was set at 640 x 640 pixels. Note that the diagrams cannot be split exactly by the patch size, and the patches cropped at the edges of each diagram overlap each other. Only the annotated symbols that appeared completely within a patch were used for training purposes. The whole drawings were annotated as opposed to the patches. An automated script was used in order to split the whole drawing into patches, and to assign the relevant annotations to each patch.

Due to the limited size of the dataset, transfer learning was used. This technique improves a learner by transferring information from one domain to another [195]. Both models were pre-trained on a large-scale object detection dataset, 2017 Microsoft COCO [191]. All model layers were fine-tuned during training.

4.4 Experiment and Results

4.4.1 Experiment Setup

The experiment can be divided into two phases. The first is the Deep Learning Model Training on Construction Symbols and second is the Method Evaluation, as shown in Figure 4.6. The input to the first phase is the pre-processed diagram dataset that was the output from the Data Preparation phase shown in Figure 4.2.

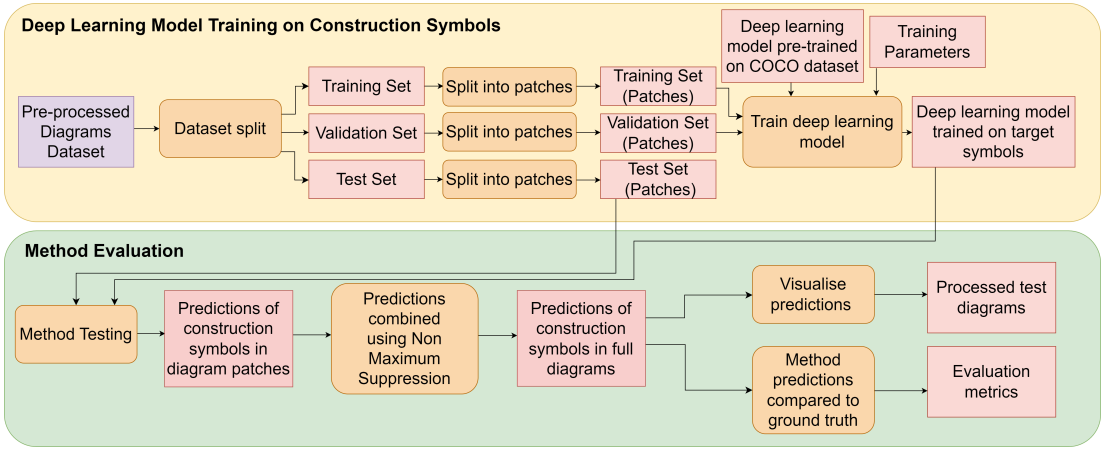


Figure 4.6: Experiment steps. Inputs are shown in purple, processes are shown in orange and outputs are shown in pink.

The pre-processed dataset of 198 diagrams was split into training, validation and test sets. These contained 168, 15 and 15 diagrams respectively. Each subset contained all three diagram types and instances of each symbol class. As the classes were unevenly distributed across the diagrams, the distribution of classes is imbalanced within each of the training, validation and test sets. It also different in each subset. This is shown in Figure 4.5.

Following the patch-based approach described above, the 168 training diagrams were split into $59,136$ patches. Of these, $1,633$ contained at least one annotated symbol. Patches not including symbols of interest were also included in the training data, in equal ratio to labelled patches. This may help to reduce false positives occurring due to similar shapes in the diagrams. To select the more cluttered patches, these were randomly sampled from those which contained over 15% black pixels. The 15 validation diagrams were split into $5,280$ patches, of which 94 were annotated. Again, patches without symbols of interest were included in equal amounts to the labelled patches.

YOLOv7 was trained using a batch size of eight as initial experiments showed improved results compared with larger batch sizes. The momentum was set to 0.937, as is standard in the implementation. To help prevent overfitting, mosaic data augmentation,

which mixes four images [22] was used with a probability of 1.0. The idea is to show extra symbol contexts to the model, and it also reduces the requirement for a large mini-batch size [22]. The probability of a left-right flip was set at 0.5, and an up/down flip was set at 0.0. The image translation factor was set at +/- 0.2.

The Faster R-CNN batch size was set at four due to memory requirements. The momentum was set at 0.9, and the probability of a horizontal flip was set at 0.5. Following the original baseline model, a ResNet-50 backbone was used [88]. Note that as the aim was to compare the methods based on the two models, the data augmentations used in each approach were kept as is standard in each implementation.

Each model was trained for 100 epochs which took 2.92 hours for the YOLO-based method, and 40.86 hours for the Faster R-CNN-based method. Note that the official implementations were used[2] [196]. The experiments were carried out using an NVIDIA Quadro RTX5000 16GB GPU with 256GB RAM.

The methods were evaluated using a test set, which contains 15 drawings split into 19,995 patches, as seen in Figure 4.6. Here an overlapping patches strategy was used to ensure all symbols fully appeared within a patch. This means that overlapping predictions can occur. NMS was used to handle this, as shown in Figure 4.7. It should be noted that the overlap threshold was set at 0.3, and the confidence threshold was set at 0.005. It is worth pointing out that the testing took 0.09 hours using the YOLO-based method, and 2.72 hours using the Faster R-CNN-based method. Note that this was for the whole test set of 15 drawings. This is significantly less than the time required for manual drawing analysis, which can take hours of work per drawing and requires subject matter specialists.

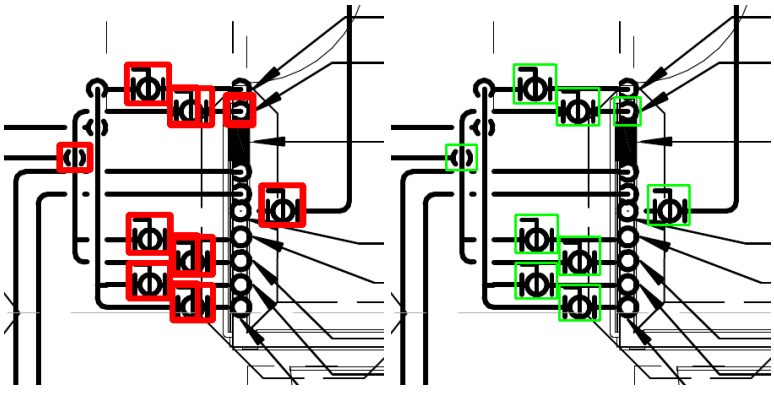


Figure 4.7: Non-Maximum Suppression was used to handle the overlapping predictions. Initial predicted bounding boxes are shown in red (left image). The results following Non-Maximum Suppression are shown in green (right image)

[2] https://github.com/WongKinYiu/yolov7

4.4.2 Evaluation Metrics

The methods were evaluated using multiple metrics, including Precision, Recall, and F1-score, refer to Section 2.2.2. The model confidence threshold was set at 0.25, which should ensure an appropriate trade-off between obtaining true positives and reducing false positives. The method was also evaluated using the mAP at IOU threshold of 0.5 (mAP@0.5). This was determined using the all-point interpolation method as in PASCAL VOC 2010 [193]. In addition, the AP@[0.5 : 0.05 : 0.95], APsmall, APmedium and APlarge were reported [191]. An open-source toolkit for object detection metrics created by Padilla et al. [192] was used to perform this calculation.

4.4.3 Results

The results were initially evaluated by visual inspection with help from domain experts in order to understand the model performance. As shown in Figures 4.8 and 4.9, this was facilitated by drawing bounding boxes around the ground truth in red, YOLO-based method correct predictions in orange and by the Faster R-CNN-based method in purple. The incorrect predictions by the YOLO-based method are shown in dark blue and those by the Faster R-CNN-based method in light blue. For in-depth analysis, the model confidence and the IOU were also shown. Note that these values are best viewed when zoomed in to Figures 4.8 and 4.9. This suggested that various symbol classes were detected well, even with multiple overlapping components. It was also observed that where a correct prediction was recorded by both methods, the difference in predicted bounding box locations was small and most visible on the larger symbols, refer to patches a and b in Figure 4.8.

Table 4.1: Method performance on the test set. The highest performing score for each metric is highlighted in bold.

Method	Accuracy	mAP@0.5	AP@[0.5:0.05:0.95]	APsmall	APmedium	APlarge
YOLO-based method	**95.8**	0.79	**0.50**	**0.27**	**0.50**	**0.68**
Faster R-CNN-based method	95.6	**0.83**	**0.50**	0.19	**0.50**	0.64

The results on the whole dataset were assessed using several metrics, as shown in Table 4.1. The mAP@0.5 of the YOLO-based method was 79%, whilst that of the Faster R-CNN-based method was 83%. Out of the 665 symbols, 637 were correctly detected by the YOLO-based method and 636 by the Faster R-CNN-based method, equivalent to an accuracy of 95.8% and 95.6%, respectively. In terms of the AP@[0.5 : 0.05 : 0.95], both methods performed equally with a score of 0.50. The results were also evaluated according to symbol size. This shows that both methods perform better the larger the symbol size is. This is likely due to more information being present in the larger symbols compared to the smaller ones. Both methods performed equally on the medium

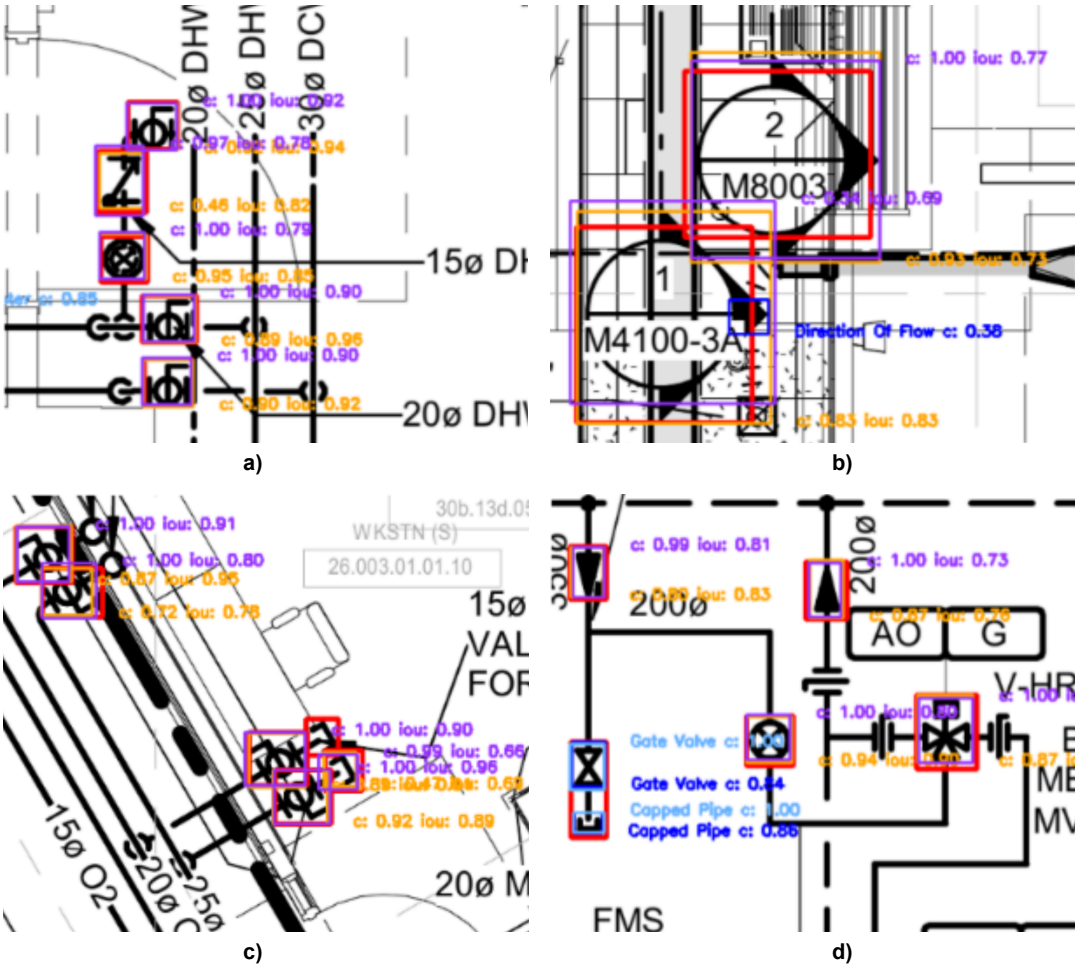


Figure 4.8: Examples of test patches. To facilitate visual inspection, bounding boxes were shown around the ground truth in red, YOLO-based method correct predictions in orange and by the Faster R-CNN-based method in purple. The incorrect predictions by the YOLO-based method are shown in dark blue and those by the Faster R-CNN-based method are in light blue. The model confidence, c, and the IOU were also shown

sized symbols, as shown in Table 4.1. It is also evident that the YOLO-based method performs slightly better on the small symbols than the Faster R-CNN-based method, with a value of 0.27 compared to 0.19 obtained for APsmall. Similarly, the YOLO-based method performs slightly better on the large symbols than the Faster R-CNN-based method, with the values of APlarge being 0.68 and 0.64 respectively. These results suggest that although the YOLO-based method outperforms the Faster R-CNN-based method on certain metrics, both methods have performed well on this challenging dataset.

The precision, recall and F1-score were calculated for each class, as can be seen in Table 4.2. These results show that both methods performed well for the detection of various

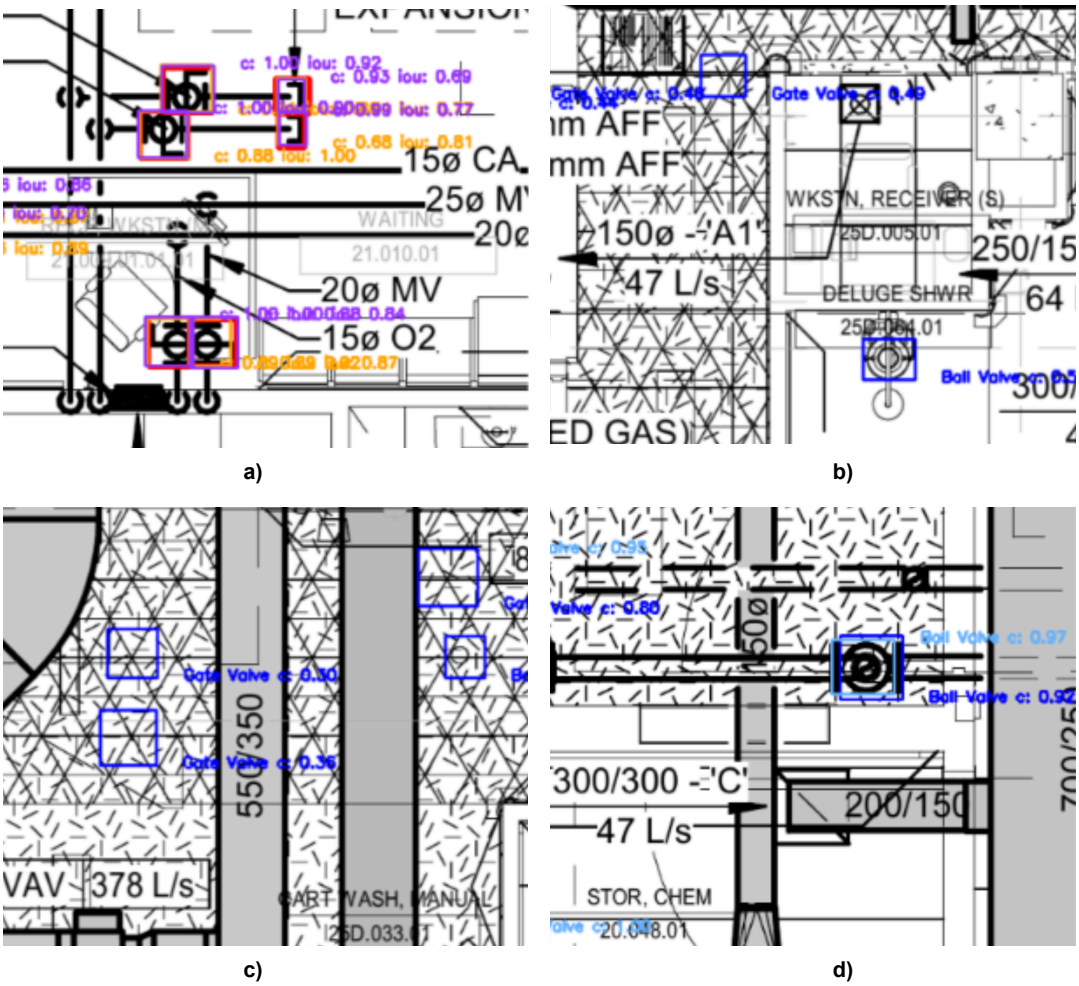


Figure 4.9: Further examples of test patches. To facilitate visual inspection, bounding boxes were shown around the ground truth in red, YOLO-based method correct predictions in orange and by the Faster R-CNN-based method in purple. The incorrect predictions by the YOLO-based method are shown in dark blue and those by the Faster R-CNN-based method are in light blue. The model confidence, c, and the IOU were also shown

symbol classes. Although class imbalance can strongly affect performance, other factors also influenced these results. For instance, the highest F1-score was not obtained for the majority class, the Ball Valve. The recall was high indicating that most instances were detected correctly. This included those in different orientations, as shown in patches a and c in Figure 4.8 and a in Figure 4.9. However, the precision was lower, which may be due to several reasons, including model bias due to class imbalance. Furthermore, similar shapes were very common, refer to the incorrect predictions of Ball Valves shown in patches b and d in Figure 4.9. The highest performance by the YOLO-based method was for the Pump and by the Faster R-CNN-based method was

Table 4.2: Method performance on the test set per class. The highest recall, precision and F1-score for each symbol is highlighted in bold.

Class	Total No.	Train No.	Test No.	YOLO			Faster R-CNN		
				Recall	Precision	F1-score	Recall	Precision	F1-score
Ball Valve	2201	2088	89	**0.99**	0.69	**0.81**	0.97	**0.70**	**0.81**
Gate Valve	1093	904	95	**1.00**	0.39	0.56	0.99	**0.90**	**0.94**
Direction Of Flow	668	430	186	**0.99**	**0.94**	**0.96**	0.98	0.89	0.93
CBV	656	577	78	**1.00**	0.77	0.87	**1.00**	**0.85**	**0.92**
Detail Legend	504	414	31	0.97	0.38	0.55	**1.00**	**0.69**	**0.82**
ACV	308	237	55	**1.00**	0.87	0.93	**1.00**	**0.98**	**0.99**
Pipe Down	279	246	21	**0.90**	**0.59**	**0.71**	**0.90**	0.58	**0.71**
Capped Pipe	279	173	59	**0.88**	**0.78**	**0.83**	0.83	0.72	0.77
Check Valve	95	71	16	0.88	0.67	0.76	**1.00**	**0.80**	**0.89**
Pump	61	44	15	**1.00**	**1.00**	**1.00**	**1.00**	0.58	0.73
V&C Provision	47	41	5	0.00	0.00	0.00	0.00	0.00	0.00
Meter	21	10	9	0.78	0.64	0.70	**1.00**	**0.82**	**0.90**
Backflow Preventer	19	12	6	0.00	0.00	0.00	**0.33**	**0.22**	**0.26**

for the ACV, even given that these were the sixth and tenth most represented classes respectively. This indicates that the results are impacted by other factors as well as class representation, such as similar shapes in the drawings.

The results also show that the lowest performance was obtained for classes with very few instances. For example, an F1-score of 0.00 was recorded by both methods for one class, the V&C Provision, which had only 47 instances. Although the Meter had fewer instances, 21, the performance was higher, likely due to the relatively consistent appearance of this symbol.

It can also be seen in Table 4.2 that there were higher levels of recall compared to precision. This was due to false positives, of which there were 320 by the YOLO-based method and 154 by the Faster R-CNN-based method. Only a few of these were as a result of the inter-class similarity. There was one prediction of a Meter as a Detail Legend by the YOLO-based method, and one prediction of a Gate Valve as a Check Valve by the Faster R-CNN-based method. The other misclassifications between target classes were that all V&C Provisions were predicted as two separate symbols, the Gate Valve and Capped Pipe, see patch d in Figure 4.8. This can be explained as the shapes that constitute the V&C Provision are essentially a combination of these two symbols, see Figure 4.3.

The majority of the false positives were due to similar shapes in the background of the drawing. The highest number of false positives for any symbol, 150, was for the Gate Valve by the YOLO-based method. This was often due to similar triangular shapes used to shade parts of the diagram, as shown in patches b and c in Figure 4.9. In contrast, the Faster R-CNN method performed better here and only predicted 9 false positives for the Gate Valve. Another noticeable difference between the methods was in the number of false positives recorded for the Detail Legend symbol, for which the YOLO-based

method predicted 47 whereas the Faster R-CNN-based method predicted less at 14. Both methods predicted a similar number of false positives for the most represented symbol in the dataset, the Ball Valve, with 39 false positives by the YOLO-based method and 36 for the Faster R-CNN-based method. Both of the methods predicted that similar shaped components in the diagram were Ball Valves, see patch d of Figure 4.9. It should also be pointed out that there were no false positives from similar shapes in the diagram border area, as this section of the drawing was removed in the drawing pre-processing, refer to Figure 4.2. Overall, these results show that both methods have high discriminative power between the target classes, and that most incorrect predictions result from the similar shapes that are used throughout the drawing.

4.5 Summary

This chapter presents a deep learning framework for the automatic processing of construction drawings. This enables symbol digitisation and can therefore automate tasks such as material takeoff. Two state-of-the-art object detection models, YOLO and Faster R-CNN, were utilised. An extensive set of experiments was carried out using a large dataset of challenging high-resolution drawings sourced from an industry partner.

The results show significant time-saving compared with manual drawing analysis. Although the highest accuracy was obtained with the YOLO-based method, both methods were shown to obtain high performance, for both recall and precision, for a range of symbols. This was obtained even with the challenges posed by the dataset, such as relatively small symbol size, different orientations and the presence of multiple overlapping objects. One limitation was that the performance was inconsistent across the classes, due to factors including the class imbalance, similar shapes and intra-class variations such as size and orientation.

Future directions of this work include investigating symbol digitisation methods that require less training data. In the following chapter, research into few-shot methods is presented. These methods have the potential to be very beneficial for engineering symbol digitisation, as they are designed to learn class representations from only a few labelled training instances.

Chapter 5

Few-Shot Symbol Detection

As was shown in Chapter 4, one of the main challenges associated with a symbol digitisation solution based on a deep learning object detector was that of data annotation. Obtaining a sufficient number of labelled training symbols is time-consuming, requires subject matter experts and is open to human error. Furthermore, for rare symbols there may be insufficient samples available in the dataset. In this chapter, a few-shot approach for the problem of symbol digitisation in EDs is presented. These methods are designed to learn from only a few labelled instances per novel class. Extensive experiments on real-world P&ID were completed, which show the method significantly improves performance compared to other state-of-the-art methods whilst requiring fewer training instances. *This work has been published in the AAI journal [39].*

5.1 Introduction

Symbol recognition is one of the main methods required for ED digitisation. Very recently, researchers have created various deep learning approaches for this purpose [44, 2, 1]. As shown in Chapter 2, these were mainly based on object detectors, such as YOLO [19, 20, 21, 22, 23, 24, 25, 26] or Faster R-CNN [88].

Object detectors typically require a large labelled training dataset, however obtaining sufficient annotated symbols is challenging and can be impossible [150]. Firstly, due to confidentiality reasons there is a lack of publicly available technical drawing datasets [197, 36], refer to Table 2.4. Secondly, the annotation task is very costly and time-consuming, potentially taking weeks for a typical dataset [2], as shown in Chapter 4. The process involves manually drawing a bounding box closely around each target symbol and then assigning the relevant class label. Given the specialist nature of the drawings, the task must be completed by subject matter experts. Thirdly, it may be

infeasible to obtain sufficient instances of the rarer classes. This can result in the class imbalance problem [33, 142] which is when a model trained on an imbalanced dataset is biased towards the majority classes.

Few-Shot Learning (FSL) is the task of learning from a limited number of training samples with supervised information [198]. The problem of Few-Shot Object Detection (FSOD) has received less attention from researchers compared to Few-Shot Classification (FSC) [199, 200]. FSOD is considered more challenging than FSC, due to the additional requirement for object localisation [150] and because multiple instances may be present per image.

5.2 Few-Shot Learning Methods

Due to the large amount of critical data trapped in undigitised EDs, there is considerable demand to automate their digitisation [6]. However, the task is considered a challenging problem, with researchers creating various methods to process these drawings over the past four decades [13, 10, 9]. Initial methods were based on traditional machine learning approaches, which demanded hand-crafted features as input [17]. Although these methods proved to be successful in specific use cases, their reliance on pre-established rules meant that they did not generalise well across the variations seen in EDs, such as morphological changes and noise [51, 52]. In recent years, deep learning-based methods have significantly improved computer vision methods for tasks such as object detection [14]. These methods outperform traditional approaches as they automatically learn features from pixel data and have improved generalisation ability.

Over the last few years, deep learning methods for symbol digitisation have been proposed. Most were based on object detection models, which predict the location and class of objects within an image. For instance, Elyan et al. [1] presented a YOLO based [21] approach for the detection of symbols in P&IDs. A symbol dataset was obtained through time-consuming manual annotation of 172 high-resolution industry P&IDs. The method performed well overall with an accuracy of 95%. However, the results were inconsistent across the symbols, with lower performance on the rare classes. Meanwhile, Jakubik [2] presented a Faster R-CNN based method for symbol detection in floor plans. To avoid extensive data labelling, they generated training data using a data augmentation technique. However, to obtain optimum accuracy, it is typically better to source the training data from the same distribution as the test data.

The need for sample-efficient symbol detection methods [44] could be addressed using FSOD. In FSOD, the object classes are split into two non-overlapping sets known as the base classes and the novel classes. Base classes have a large number of labelled

samples, while the novel classes have only a few. FSOD methods aim to transfer generic object knowledge from the common heavy-tailed objects to the novel long-tailed object categories [198].

FSOD is an active research area, with the majority of the methods being published in the last four years [201]. The models are typically based on object detection architectures, the most common being Faster R-CNN. The methods can be categorised as fine-tuning based, such as the approach introduced by Wang et al. [199], or meta-learning based, such as the method created by Kang et al. [202]. In this experiment, the methods are based on the fine-tuning approach due to the reported improved performance [203].

Wang et al. [199] introduced the frustratingly simple few-shot object detection method, and showed that good FSOD performance could be achieved using a Two-stage Fine-tuning Approach (TFA). In the first stage the model was trained using base classes and in the second stage, it was fine-tuned using all classes. Here the box classifier and box regressor were fine-tuned, and the other model components were frozen. The authors showed that their approach outperformed various methods including the meta-learning approach Few Shot Object Detection via Feature Reweighting [202].

FSOD methods based on TFA [199] have been proposed [201]. For instance, Kaul et al. [204] showed that fine-tuning the Region Proposal Network (RPN) using 30 shots of novel classes substantially increased the average recall compared to that using the base RPN. To further improve the performance, the ROI module was fine-tuned. They incorporated semi-supervised learning to obtain additional samples of novel classes and reduce the class imbalance, however this relies on additional novel class data being available. Meanwhile, Fan et al. [200] found that the RPN in TFA was not class-agnostic and was instead biased towards the base classes. This suggests that allowing the RPN and ROI to learn from novel class data in the fine-tuning phase may improve the performance.

The most commonly used FSOD benchmarks are the splits introduced by Kang et al. [202] on PASCAL VOC [95] and COCO [191] datasets. Although these datasets were widely used in FSOD, these datasets do not represent realistic rare categories and further research on more realistic datasets is needed [201].

The literature shows that most engineering symbol digitisation methods were based on object detection models that typically require a large labelled training dataset. However, this can be infeasible to acquire due to data unavailability, rare symbols and the costly annotation process. It was also seen that few-shot object detection is a relatively new research field in which methods are designed to learn from limited data,

however they have not yet been explored for engineering symbol digitisation.

5.3 Methods

In this section, firstly the dataset of real-world engineering diagrams is introduced. Next, the methods used to pre-process this data are described. This is followed by a detailed description of the few-shot symbol detection method.

5.3.1 Dataset

A dataset of 172 P&IDs was sourced from an industry partner in the oil and gas domain. This is the same proprietary dataset that was used in Chapter 3. The images were converted from PDFs to high-resolution PNG files with a size of $7,428$ x $5,251$ pixels. The diagrams represent various engineering equipment and their connections. For experiment purposes 25 symbol classes, such as valves and flow labels, were selected, as shown in Figure 5.1.

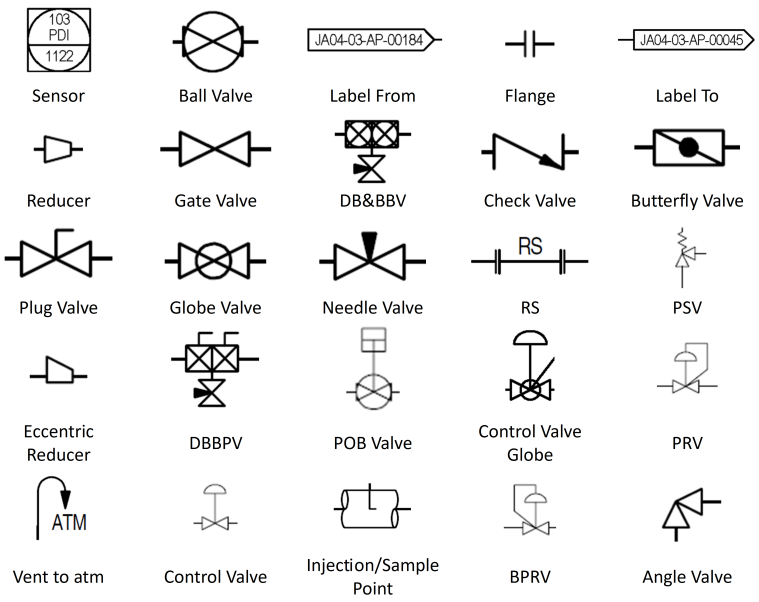


Figure 5.1: The P&ID symbol classes used in the experiment. These are challenging to detect as they are represented by only a few shapes, have high inter-class similarity and high intra-class variability.

The dataset is very challenging for object detection models. One reason for this is that the symbols are only represented by a few lines and therefore contain few features for a model to learn from. Additionally, there is usually high intra-class variability, high inter-class similarity, and the presence of similar shapes. Furthermore, unlike commonly used datasets such as PASCAL VOC, in which objects are mostly of the

same orientation and are mainly located in the center of images [205], engineering symbols are frequently in different orientations and can be located anywhere in an image.

The diagrams had been manually annotated to obtain a labelled symbol dataset. Various annotation software is available for this task, such as Sloth[1] and Computer Vision Annotation Tool (CVAT)[2]. Sloth was used here. The task is known to be very time-consuming and costly for EDs [29].

5.3.2 Data Pre-Processing

A series of image processing algorithms was used to remove the diagram border. This section contained various information, such as the drawing title and drawing revision details, however it contained no equipment symbols. First, the diagram was binarised and a CC algorithm was used to locate the largest CC of white pixels. This CC was considered to be the background of the main diagram area, as in Section 4.3.2. The pixels were considered as connected if they had four-way connectivity. Each pixel outwith the bounding box of the largest CC was then replaced with a white pixel using an image mask.

The diagrams are significantly larger than the typical input size for neural networks, and therefore a patch-based method [194, 1] was used. Here, the patch size was 640 x 640 pixels. It should be noted that the patches overlapped each other at the diagram edges. Any annotation that overlapped multiple patches was not used in the training data.

5.3.3 Few-Shot Symbol Detection

The main idea of the few-shot approach used is to separate learning of different model components. The method is based on TFA [199], which separates feature representation learning and box predictor learning. The model architecture was based on Faster R-CNN [88]. The feature extractor components consist of a ResNet-101 [96] with FPN [170] backbone, RPN, ROI pooling layer and ROI feature extractor. The box predictor consists of a box classifier and box regressor, which predict the object categories and bounding box regression offsets respectively. The model was trained in two stages, as shown in Figure 5.2.

The first is base training, in which the entire object detector was trained on the base classes. The second stage is few-shot fine-tuning. In TFA the last layers of the model,

[1] https://sloth.readthedocs.io/en/latest/
[2] https://github.com/opencv/cvat

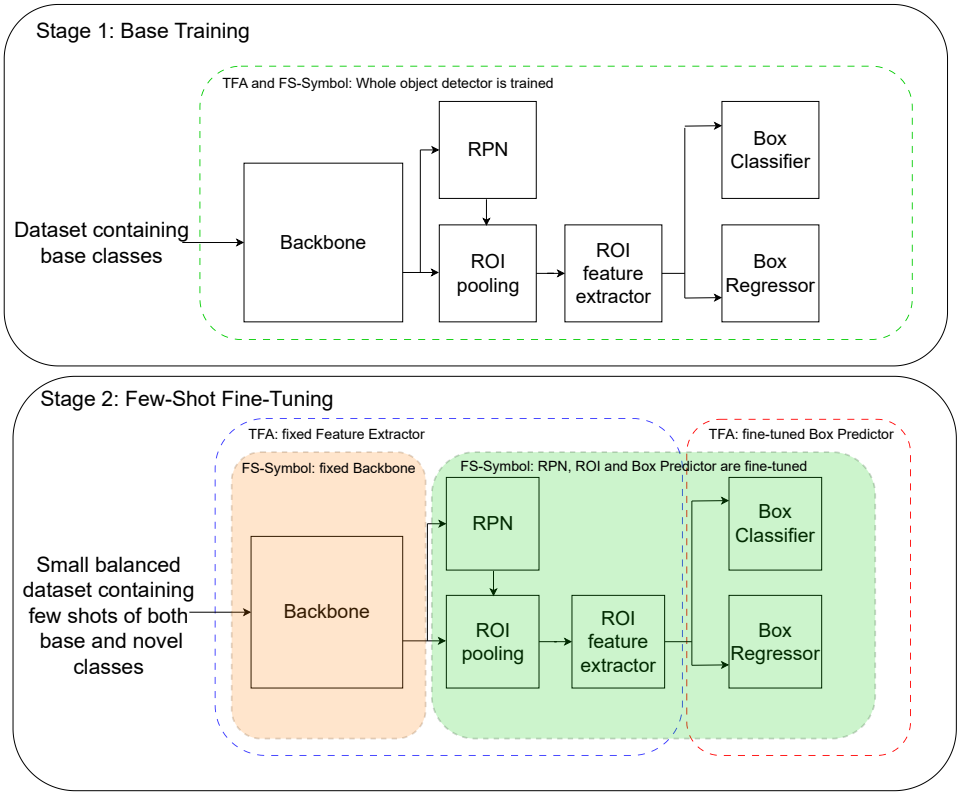


Figure 5.2: The TFA [199] and FS-Symbol methods. In the first stage, the whole object detector is trained on the data-abundant base classes. In the second stage of TFA, the feature extractor is fixed and the box predictor is fine-tuned on a small balanced dataset containing few shots of base and novel classes. In the second stage of FS-Symbol, the backbone is frozen and all other model components are fine-tuned.

the box classifier and regressor, were fine-tuned while the feature extractor was fixed. In Few-Shot-Symbol (FS-Symbol), TFA was altered with the aim to improve the performance on the novel classes. To do this, the RPN and ROI were unfrozen in the second stage, as can be seen in Figure 5.2. In the second step, a small balanced support dataset consisting of K shots of both the base and novel classes was used for training.

In both stages, the model was trained using the multi-task loss function as shown in Equation 5.1.

$$\mathcal{L} = \mathcal{L}_{rpn} + \mathcal{L}_{cls} + \mathcal{L}_{loc} \tag{5.1}$$

Here $\mathcal{L}_{rpn}$ is the RPN loss, which is the object proposal loss that determines the foreground from background and refines the anchors. $\mathcal{L}_{cls}$ is the cross-entropy loss for the box classifier and $\mathcal{L}_{loc}$ is the smoothed L_1 loss for the box regressor.

In place of fully connected classification layers, a cosine similarity classifier based on the instance-level distance measurement was used. The classifier outputs scaled similarity scores where the similarity score, $s_{i,j}$ between the i-th object proposal of the input x, and w_j, the weight vector of class j, is defined as shown in Equation 5.2. Here, α is a scaling parameter set to 20 and $\mathcal{F}(x)_i$ is the input feature of the i-th object.

$$s_{i,j} = \frac{\alpha \mathcal{F}(x)_i^T w_j}{||\mathcal{F}(x)_i|| \, ||w_j||} \tag{5.2}$$

5.4 Experiment and Results

Six few-shot methods were used. The experiment setup is discussed in detail here. Evaluation metrics related to few-shot detection are also introduced. This section also includes the presentation of the results and detailed analysis.

5.4.1 Experiment Setup

The dataset of 172 P&IDs was split into training and test sets, which contained 155 and 16 diagrams respectively. Note that one diagram was mislabelled and therefore not used. No validation set was used due to the limited instances available for the minority classes.

The patch-based approach detailed in Section 5.3.2 was then used to split the 155 training diagrams into 16,488 patches. Only those patches labelled with one or more symbols, 3,984 patches, were included in the training dataset. Using the same method, the test diagrams were split into 5,888 patches. Here, a patch overlap of 320 pixels, larger than the maximum symbol dimension, was used to ensure that all pixels were fully contained within a patch.

Following the FSOD setting, the symbol classes, C, were split into base classes C_{base} and novel classes C_{novel} such that $C_{base} \cap C_{novel} = \varnothing$. The 7 least represented symbols were chosen as the novel classes and the remaining 18 symbols as the base classes. This results in a base to novel class ratio of 2.6:1, which is similar to the 3:1 ratio used in the common FSOD benchmarks [202, 199].

The model was trained using a batch size of 8 instead of the default 16. The linear scaling rule [206], which states that the learning rate should be multiplied by k when the minibatch size is multiplied by k, was used to set the learning rate to 0.01 in base training and 0.0005 in fine-tuning. The model was trained for 17 epochs in base training and 3200 epochs in fine-tuning, following the settings in TFA [199]. There were 200 warmup iterations. The novel class weights for the box prediction networks were

randomly initialised prior to fine-tuning. Multiscale training was used to improve model performance on symbols of different sizes. Here the patch size, x, was selected such that $x \in \{480, 512, 544, 576, 608, 640, 672, 704, 736, 768, 800\}$. The probability of horizontal flip was set to zero, to ensure that the high amount of text-containing symbols remain realistic.

The number of shots, K, of novel classes was set to $K = 1, 2, 3, 5, 9$. Note that this is similar to the setting in the PASCAL VOC benchmark, however here the maximum K value was set at nine as there were insufficient instances of each class in the diagram dataset to use ten shots. To ensure fair comparison between the all the model architectures, the K shots were randomly selected and fixed across the experiments.

Six methods were evaluated, as shown in Table 5.1. The first, TFA (R-101), was the baseline few-shot method TFA [199] with a ResNet-101 [96] backbone. In the second method, TFA (R-50), a smaller network, ResNet-50 [96], was used as the backbone. Both networks were pre-trained using ImageNet [207], which is a large-scale dataset designed for image classification.

Table 5.1: Method training settings in the fine-tuning phase

Method	**Fine-tuned components**			
	Backbone	**RPN**	**ROI**	**Box Predictor**
TFA (R-101) [199]				✓
TFA (R-50)				✓
Balanced	-	-	-	-
Few-Shot (FT all)	✓	✓	✓	✓
Few-Shot (FT ROI + box)			✓	✓
FS-Symbol		✓	✓	✓

In the third method, Balanced, an undersampled training set comprising of a few shots for all 25 classes was used. By this definition, all classes are considered novel and there are no base classes. Here the balanced dataset of K shots was used to train the whole model, similar to the base training step shown in Figure 5.2. No fine-tuning phase was used. The balanced models were trained for 100 epochs.

Three other methods were evaluated to determine the impact of unfreezing specific model components in the fine-tuning stage, as shown in Table 5.1. In the first of these methods, Few-Shot (FT all), the entire model was unfrozen and fine-tuned. Next, in Few-Shot (FT ROI + box), the backbone and RPN were frozen whilst the ROI and box predictor components were fine-tuned. Lastly, in the proposed method, FS-Symbol, the model backbone was frozen and all other components were fine-tuned.

The inference was carried out on individual test patches, and the results were combined. Non-Maximum Suppression was used to handle the overlapping predictions that occurred as a result of the patches strategy.

Table 5.2: Few-shot detection performance on the test diagrams for novel symbols (nAP), base symbols (bAP) and all symbols (mAP). Highest performance at each shot is in bold.

Method / Shot	nAP					bAP					mAP				
	1	**2**	**3**	**5**	**9**	**1**	**2**	**3**	**5**	**9**	**1**	**2**	**3**	**5**	**9**
TFA (R-101) [199]	10.0	8.4	27.6	24.4	43.0	**97.3**	**97.8**	97.6	**98.3**	**98.2**	72.9	72.8	78.0	77.6	82.8
TFA (R-50)	1.5	15.4	22.0	24.7	29.7	97.2	97.3	97.5	97.6	97.6	70.4	74.4	76.4	77.2	78.6
Balanced	38.8	39.3	67.7	68.9	82.8	29.6	37.3	43.6	47.0	56.6	32.2	37.9	50.3	53.2	64.0
Few-Shot (FT all)	38.6	33.5	62.9	69.3	71.3	48.1	57.0	58.7	59.6	64.4	45.4	50.4	59.9	62.3	66.3
Few-Shot (FT ROI + box)	45.0	**44.1**	54.2	61.7	65.2	96.1	96.2	**98.0**	96.9	98.1	**81.8**	**81.6**	**85.7**	**87.0**	**88.9**
FS-Symbol	**45.1**	42.8	**68.2**	**74.8**	**83.4**	78.8	85.9	87.1	87.9	89.9	69.4	73.8	81.8	84.2	88.0

The methods were based on the official TFA implementation[3]. All experiments were carried out using a NVIDIA Quadro RTX5000 16GB GPU with 256GB RAM.

5.4.2 Few-Shot Detection Evaluation Metrics

Object detection models are commonly evaluated using the mAP, see Section 2.2.2. In FSOD, separate metrics are used for the base classes AP (bAP) and novel classes AP (nAP). Here, mAP, bAP and nAP were all used. The AP values were calculated using the same method as described in Chapter 4. The methods were also evaluated on a per-class basis using the recall, refer to Section 2.2.2.

5.4.3 Results and Discussion

The various few-shot methods were evaluated at each K value using the nAP, bAP and mAP, as presented in Table 5.2. The results show that the performance typically improves with the K value, however in certain cases increasing K results in a slight performance decrease. Note that there is a large amount of intra-class variance, and this finding is likely to be related to how closely the few randomly selected training instances represent the test data. Comparing the results of the first two methods shows that using ResNet-50 (R-50) instead of ResNet-101 (R-101) for the model backbone negatively impacted performance across all metrics in most cases. This suggests that the additional complexity of the R-101 network improves the model's ability to capture the symbol features. An R-101 backbone was therefore used in all further experiments.

The results also show that FS-Symbol outperforms all other methods for nAP at most K values ($K = 1, 3, 5, 9$). At $K = 2$, FS-Symbol performance is the second highest, whilst freezing the RPN in the second training phase, Few-Shot (FT ROI + box), resulted in the highest nAP by 1.3. Across all shots, there was an increase of between 35.1 and 50.4 in nAP using FS-Symbol compared to the baseline TFA (R-101) method. A statistical test, the t-test, was carried out to determine if the difference in novel class performance using FS-Symbol compared to the baseline TFA (R-101) method was significant. A

[3] https://github.com/ucbdrive/few-shot-object-detection

p-value of 0.0045 was obtained, which is less than the alpha value of 0.05 and therefore shows a statistically significant improvement. This suggests that allowing the RPN and ROI to learn from novel symbol data improves the region proposals, resulting in higher performance.

The highest base class performance was obtained using the baseline TFA (R-101) method for $K = 1, 2, 5, 9$. Fine-tuning the ROI in addition to the box predictor, Few-Shot (FT ROI + box), gave the highest performance at $K = 3$ and resulted in a small decrease of up to 1.6 bAP at other K values. These results highlight that fine-tuning the model backbone, RPN and ROI can lead to a loss of information learned in the first training stage.

The highest mAP at all shots was recorded using the Few-Shot (FT ROI + box) method. There was a statistically significant improvement using this method compared to the baseline TFA (R-101) method, as the *p-value* obtained using the t-test was 0.008. It was also observed that using the Balanced method harmed the bAP and thus the mAP, compared to using all available base class instances. Although there was no class imbalance, the performance was inconsistent across the various classes. This highlights that additional challenges exist for symbol detection, which includes high intra-class variation, high inter-class similarity and varying symbol orientation.

Another important metric to consider for engineering symbol detection is the recall, which measures the fraction of symbols correctly detected, see Equation 2.2. The per-class recall, or accuracy, obtained using various few-shot methods and a YOLO-based method [1] trained on the fully annotated dataset was compared, as shown in Table 5.3. Note that more base class training samples were used for the YOLO method [1] as a result of the larger image patch size, 1250 x 1300 pixels compared to 640 x 640 pixels.

The results in Table 5.3 clearly show that the highest recall for each novel class was recorded using the FS-Symbol method. For five of the seven novel classes, a recall of 1.00 was obtained using only nine training samples per class. Furthermore, the recall for the four least represented classes increased considerably compared to that obtained with the YOLO-based method. For instance, for the Vent to atm symbol, a recall of 0.88 was recorded using FS-Symbol compared to 0.25 using the YOLO-based method. Note that nine samples of this symbol were used to train the few-shot methods whereas nineteen were used to train YOLO. Also evident is the need to preserve features learned from the data-abundant base classes, which is shown by analysing the performance of the Few-Shot (FT all) method. For instance, a recall of 0.00 was obtained for the novel class Angle Valve, compared to 0.50 with TFA (R-101). This suggests that fine-tuning

Table 5.3: Comparison of few-shot and object detection method recall on test diagrams. Few-Shot 1 is Few-Shot (FT all) and Few-Shot 2 is Few-Shot (FT ROI + Box). Few-shot results reported using $K = 9$. Highest performance for each class is in bold.

Class	Test No.	YOLO Train No.	Few-Shot Base Train No.	Recall						
				YOLO [1]	TFA [199]	TFA (R-50)	Balanced	Few-Shot 1	Few-Shot 2	FS-Symbol
Sensor	302	2810	1739	0.98	0.97	**0.99**	0.10	0.15	0.98	0.43
Ball Valve	213	1629	1346	**0.99**	0.46	0.40	0.18	0.21	0.44	0.39
Label From	103	1347	982	**1.00**	0.92	0.94	0.35	0.39	0.92	0.58
Label To	113	1178	828	**1.00**	**1.00**	0.99	0.35	0.50	0.99	0.80
Flange	158	1110	739	0.77	**0.99**	0.98	0.33	0.25	0.98	0.58
Reducer	91	821	505	0.99	**1.00**	**1.00**	0.68	0.70	**1.00**	0.77
DB&BBV	67	542	469	**0.98**	0.96	0.96	0.36	0.31	0.96	0.58
Gate Valve	110	535	429	0.94	**1.00**	**1.00**	0.51	0.61	**1.00**	0.91
Check Valve	42	396	335	**1.00**	**1.00**	**1.00**	0.40	0.38	**1.00**	0.74
TOB/Butterfly Valve	59	178	168	0.98	**1.00**	**1.00**	0.38	0.62	**1.00**	**1.00**
Plug Valve	8	173	154	**1.00**	**1.00**	**1.00**	0.80	0.81	**1.00**	**1.00**
Globe Valve	7	161	150	**1.00**	**1.00**	**1.00**	**1.00**	0.90	**1.00**	**1.00**
Needle Valve	10	160	133	**1.00**	**1.00**	**1.00**	0.71	0.86	**1.00**	**1.00**
RS	26	143	114	**0.92**	0.88	**0.92**	0.65	0.77	0.88	0.85
PSV	25	118	94	**0.88**	0.74	0.78	0.17	0.39	0.83	0.39
Eccentric Reducer	23	98	92	0.96	**1.00**	0.88	0.76	0.84	**1.00**	0.88
POB Valve	16	84	65	**1.00**	0.94	0.94	0.69	0.62	0.94	0.94
DBBPV	15	83	65	**1.00**	0.93	**1.00**	0.87	0.93	0.93	**1.00**
PRV	8	32	0	**1.00**	0.83	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
Control Valve Globe	6	30	0	**1.00**	0.88	0.88	**1.00**	**1.00**	0.62	**1.00**
Control Valve	5	22	0	**1.00**	**1.00**	0.00	**1.00**	**1.00**	**1.00**	**1.00**
Vent to atm	8	19	0	0.25	0.00	0.00	0.62	0.50	**0.88**	**0.88**
Injection/Sample Point	2	13	0	0.50	0.00	0.00	**1.00**	**1.00**	0.50	**1.00**
Angle Valve	2	11	0	0.00	**0.50**	**0.50**	**0.50**	0.00	**0.50**	**0.50**
BPRV	5	11	0	0.00	0.20	0.00	**1.00**	**1.00**	**1.00**	**1.00**

the whole model results in a loss of information learned in the first training stage.

Further evidence for the validity of the FS-Symbol training approach can be seen by comparing the results obtained with those using TFA (R-101). For example, the latter method did not detect the classes Vent to atm and Injection/Sample Point, however recall values of 0.88 and 1.00 were recorded using FS-Symbol. These are the only two novel classes that are not valves and as such they are more visually distinct from the base classes compared to the other novel classes, refer to Figure 5.1. This indicates that fine-tuning only the box predictor on novel class data may not be sufficient when there is a large difference in symbol appearance between the novel and base classes.

Base class performance was typically higher using the YOLO-based method compared to the FS-Symbol method. There were only five classes for which equal or higher recall was recorded using the latter method. Another finding is that for certain base classes, competitive recall was recorded using several of the few-shot methods compared to the YOLO-based method. For example, for the Sensor symbol, a recall of 0.98 was recorded using the YOLO-based method, compared to 0.97, 0.99 and 0.98 using TFA (R-101), TFA (R-50) and Few Shot (FT ROI + box), respectively.

The performance of FS-Symbol compared to TFA (R-101) can also be observed in Figure 5.3. In these processed test patches, the ground truth bounding boxes are shown in red, correct predictions in orange and incorrect predictions in green and blue. These patches show the improved performance of FS-Symbol on the novel classes. For example, in Figure 5.3 d, the Vent to atm symbols shown were correctly predicted by FS-Symbol but not the TFA (R-101) method. Figure 5.3 g shows a BPRV symbol that

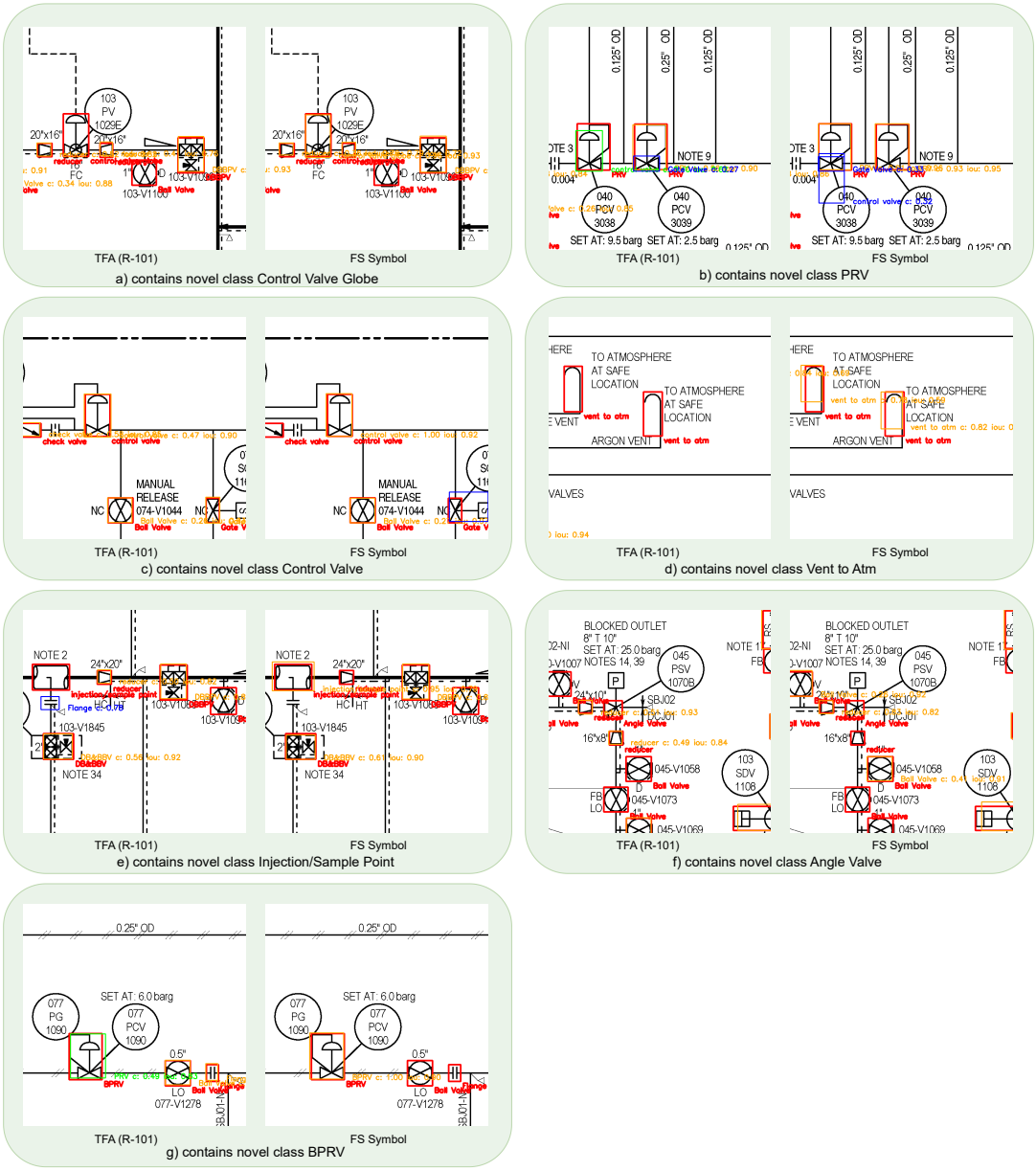


Figure 5.3: Small sections of test diagrams. In each image pair, the left image was processed using TFA (R-101), and the right image was processed using FS-Symbol. Both methods used $K = 9$. Ground truth bounding boxes are shown in red, correct predictions in orange and incorrect predictions in green and blue. The confidence and IOU are also shown.

was successfully detected using FS-Symbol, but predicted as a PRV by TFA (R-101). As these two classes contain the same shapes but with different orientations, see Figure 5.1, this suggests that training the RPN and ROI on novel data has improved the method's discriminative ability between similar symbols. The test patches also indicate that

base class performance was higher using TFA (R-101) compared to FS-Symbol. For example, they contain instances of Reducer, Ball Valve and Flange symbols detected by the first method but not the latter, see Figure 5.3 e and g. Overall, these results suggest that the training approach implemented in FS-Symbol improves the model's ability to detect novel classes.

5.5 Summary

In this chapter, one of the first approaches to the problem of few-shot symbol detection in EDs is presented. The method can be used to detect rare classes using fewer than ten training samples. Furthermore, this approach allows new symbols to be incorporated into an existing object detector without extensive annotation. Thorough experiments on complex EDs sourced from industry were completed to demonstrate the validity of the proposed method.

Various few-shot methods were evaluated and the results show that the highest performance on the novel classes was obtained using the proposed approach. Statistically significant improvement compared to the baseline few-shot method was also shown. The method was also compared against an object detection-based method trained on a dataset of fully annotated diagrams, and improved novel class performance was reported. The research also showed limitations of few-shot methods for symbol digitisation, with the main drawback being a reduction in performance for the majority of the base classes.

A future research direction is to create methods that obtain competitive performance on both the novel and base classes. This could potentially be achieved using an ensemble method that combines object detection and few-shot models. Overall this chapter opens up a new direction towards using few-shot approaches for engineering symbol digitisation, which is highly beneficial for rare symbols and also reduces the required annotation effort.

It has been shown here that engineering symbol datasets are inherently imbalanced, which can result in deep learning detectors that are biased towards the overrepresented symbols. Another related area where class imbalance is a known issue is symbol classification [31]. This problem is the focus of the following chapter, which presents a method to improve multiclass imbalanced classification of engineering symbol datasets.

Chapter 6

Multiclass Imbalanced Symbols Classification

It is evident from Chapters 4 and 5, that class imbalance is a problem in engineering symbol datasets. It has also been shown in Chapter 2, that one of the barriers to the development of deep learning solutions for ED digitisation is a lack of publicly available datasets. This chapter aims to address these issues by firstly, presenting a dataset of symbols extracted from real-world EDs and secondly, providing a method for handling multiclass imbalanced classification in this challenging scenario. *This work was presented at the 2024 ICDAR [40].*

6.1 Introduction

Class imbalance is a research challenge in ED digitisation [27]. Class imbalance occurs when one or more classes is over represented or underrepresented in a dataset. Typically, supervised learning models trained on imbalanced datasets are biased towards the majority class [208], leading to missclassifications of minority samples as majority samples [209]. Research into class imbalance has predominantly focused on binary classification rather than the multiclass scenario [210]. Additionally, although extensive research has been carried out on class imbalance in traditional machine learning, less research has considered the problem in deep learning [142, 33].

Another challenge in ED digitisation is a lack of publicly available annotated engineering symbol datasets [13]. A well-defined labelled symbol dataset was identified as a requirement to fully benefit from deep learning models for ED digitisation. It should be noted that acquiring a representative symbols dataset through manual symbol annotation is a significant task, considering a P&ID can contain in excess of 100 symbols

of multiple types. An additional consideration in obtaining a symbol dataset relevant to multiple sets of P&IDs, is that multiple drawing standards are used for equipment symbols representation [13].

In this chapter, the two research challenges of lack of ED datasets and multiclass imbalance are addressed. A new multiclass symbol dataset is presented to further research in this important area. The dataset comprises 7,728 symbols distributed across 48 classes from two P&ID drawing standards. The dataset contains symbols extracted from a set of P&IDs using an object detection method [1] and symbols from the Symbols in Engineering Drawings (SiED) dataset [31]. In addition, this chapter also presents a technique to handle multiclass imbalanced data classification. The technique is extended from CDSMOTE [189], which handles class imbalance in binary datasets. CDSMOTE uses class decomposition to reduce dominance of the majority class and synthetic oversampling to increase representation of the minority class. The multiclass imbalance data method uses class decomposition and involves synthetic oversampling of multiple minority classes to rebalance the dataset. CNN classification experiments are also presented.

6.2 Class Imbalance

Existing literature highlighted that uneven data distribution is inherent in P&IDs [27], meaning that class imbalance is a problem in this domain. Class imbalance occurs when classes are not approximately equally represented in a dataset [211]. In an imbalanced binary dataset, the minority (positive) class is underrepresented compared to the majority (negative) class. Multiclass imbalanced datasets consist of more than two classes, in which the majority class(es) is over represented compared to the minority class(es). Imbalanced datasets were reported as a challenge in obtaining accurate deep learning models [111, 208] and are known to lead to deep learning models biased towards majority classes [189].

Class imbalance in EDs has been discussed in published literature [31, 1, 111]. The SiED dataset was presented in [31]. To obtain the dataset, symbols were extracted from P&IDs using a combination of interactive and traditional image processing methods. The method extracted 2432 symbols from 39 symbol classes, however the method required fine tuning to extract symbols with any change in representation. Classification experiments using a CNN model showed that underrepresented classes in the dataset recorded comparatively lower classification performance compared to overrepresented classes. The class imbalance problem was also observed in engineering symbol detection [1]. In [1], whilst good performance was observed overall, a significantly lower class accuracy was obtained for the symbols that were underrepresented in the training

dataset.

A range of methods address class imbalance by introducing new data samples. These methods oversample minority classes by adding synthetic samples, instead of resampling instances from the original dataset. One popular synthetic oversampling method is the Synthetic Minority Oversampling Technique (SMOTE) [211]. SMOTE is designed to create artificial data based on similarities between existing minority samples. The method generates an artificial sample by interpolating between a minority class sample and one of its k-nearest neighbours [211]. SMOTE creates a broader decision region for the minority class, compared to oversampling original minority class instances [211].

CDSMOTE [189] is a technique to handle class imbalance in binary datasets. The technique reduces the impact from the original majority class while retaining all available information for training, unlike other undersampling approaches. In the technique, the k-means clustering algorithm is used to decompose the majority class into smaller sub-classes. To increase minority class representation, the minority class is then oversampled using SMOTE.

6.3 Methodology

6.3.1 Dataset

The dataset consists of P&ID symbols obtained from two different drawing standards. An object detection method extracted 5,296 symbols representing 23 classes from one drawing standard. To represent a different drawing standard, $2,432$ symbols from 39 symbol classes were acquired from the SiED dataset [31]. In total the dataset contains $7,728$ instances representing 48 symbol classes. To extract the symbols from the first drawing standard, a set of 137 P&IDs was obtained from an industry partner. An object detection method based on YOLOv3 [21] was developed. The method localises most symbols in the diagrams, for additional details the reader is referred to [1]. Following object detection, a post-processing step verified extracted regions as true or false positives. Each extracted region was considered a true positive if it contained the whole engineering symbol. Partial symbol detections were considered as false positives. The missclassified symbols were reassigned with the correct class. The method resulted in the extraction of $5,296$ symbols representing 23 classes. The dataset has been made available [1].

To increase the applicability of the symbol dataset across a range of P&ID drawing standards, the symbols extracted using the object detection method [1] were combined

[1] https://github.com/carlosfmorenog/CDSMOTE-NONBIN-Symbols

with the SiED dataset [31]. Symbol class names are noted as Type 1 or Type 2 where the representation of the symbol varied in the two different symbol standards. Each symbol in the dataset is a row vector of 100×100 features which are pixel values. One instance of each class is presented in Figure 6.1.

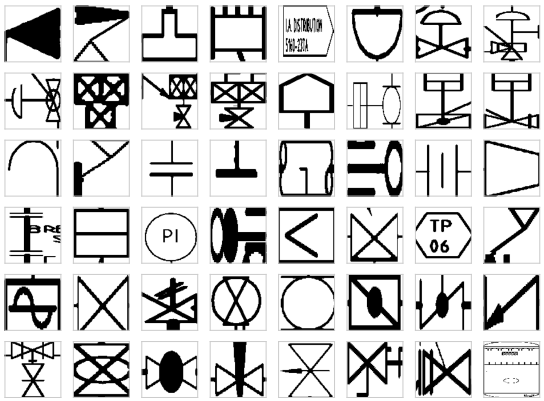


Figure 6.1: One instance of each class in the dataset.

The class distribution within the dataset is highly imbalanced, as observed in Figure 6.2. The majority class, *Sensor*, contains 2,845 instances and represents 36.8% of the dataset. In comparison, there are 3 classes, (*Barred Tee*, *Ultrasonic Flow Meter* and *Valve Butterfly Type 2*), that are each represented by one instance, and each represents 0.01% of the total dataset.

In total 60% of the entire dataset consists of only 3 of the 48 classes present in the dataset. To quantify the level of imbalance in the dataset, the imbalance ratio ir and fraction of minority classes fm are used as defined in [33] and [212].

The imbalance ratio, ir, is defined in Equation 6.1:

$$ir = \frac{max_i M}{min_i m} \tag{6.1}$$

where $max_i M$ is the maximum number of instances in a majority class and $min_i m$ is the minimum number of instances in a minority class. The fraction of minority classes, fm is defined in Equation 6.2:

$$fm = \frac{mt}{t} \tag{6.2}$$

where mt is the total number of minority classes and t is the total number of classes in a dataset. The imbalance ratio ir of the symbol dataset is 1422.5. The fraction of

minority classes, fm, is 0.978, which reflects the presence of multiple minority classes in the dataset. Considering the low absolute sizes of multiple minority classes, undersampling to obtain a fully balanced dataset would significantly reduce the amount of data available for learning. Therefore, an approach that incorporated oversampling as opposed to undersampling was used.

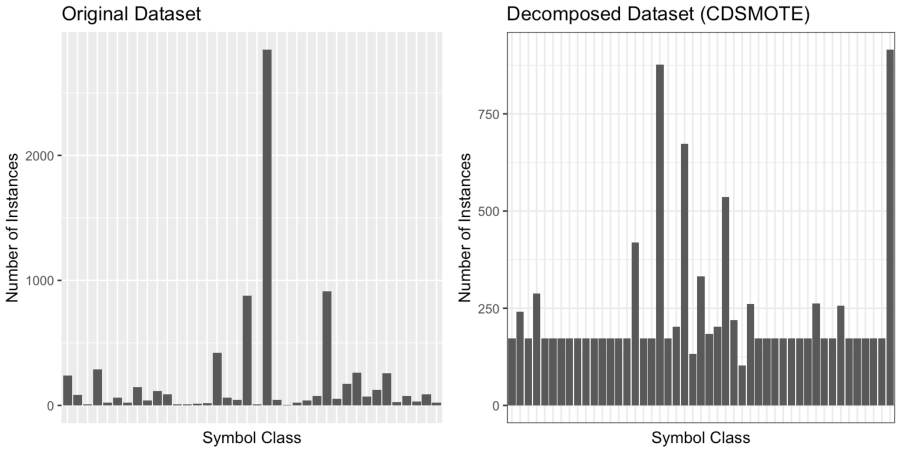


Figure 6.2: Class distribution in the original dataset (left) and the decomposed (new) dataset (right). Note that y-axes have different scales on the two subplots.

6.3.2 Multiclass Imbalance Handling Method

To address class imbalance, data resampling methods which undersample majority classes and/or oversample minority classes can be used. In the case of undersampling, information is lost from the dataset. To avoid this, the proposed multiclass method is extended from CDSMOTE [189], which is a method for handling class imbalance in binary datasets, that adjusts the data distribution whilst avoiding information loss. CDSMOTE uses class decomposition to decompose the majority class into sub-classes using the k-means clustering algorithm. Afterwards, oversampling by means of SMOTE [211] is applied to the remaining classes to increase representation of the minority classes. In the multiclass variant of this method, the majority class (i.e. *Sensor*) is decomposed into multiple sub-classes given the vast imbalance ratio with respect to the other classes. Then, synthetic oversampling using SMOTE [211] is applied to multiple minority classes with the aim of rebalancing the data distribution.

Due to the use of SMOTE as the oversampling algorithm, both the binary and multiclass versions of CDSMOTE require at least 5 instances per class for the synthetic oversampling. Therefore, classes that contained fewer than 5 instances were excluded from the dataset. To apply CDSMOTE in a multiclass scenario, we first located the

majority class in the dataset (i.e. *Sensors* with 2845 samples). Afterwards, we decomposed this class into 10 different subclasses, given that this way we could generate a better spread of samples amongst classes. Although other classes e.g. *Valve* still presented considerable majority over the rest (i.e. 915 samples), we did not decompose those as we simply wanted to test whether by decomposing the majority class would suffice to improve results. Afterwards, we calculated the new average number of samples per class (i.e. 173) and applied SMOTE to all classes with fewer samples than this value to generate synthetic samples. Notice that although the use of SMOTE is typically reserved for tabular data rather than image data, an inspection of the obtained samples showed that the quality of the synthetic samples is acceptable given that symbols have a definite shape which is easy to replicate by synthetic generation methods, refer to Figures 6.3 and 6.4.

6.3.3 Classification Method

A CNN was used for evaluation of classification performance of the imbalanced symbol dataset. A CNN [17] was chosen for classification experiments as significant advancements in the field of image classification have been obtained using CNNs in recent years [213]. The CNN used had the following model architecture: 100×100 convolutional layer with 32 filters; 2×2 max pooling layer; two convolutional layers with 64 filters; 2×2 max pooling layer; two convolutional layers with 128 filters; 2×2 max pooling layer; two fully connected layers; softmax output layer. The number of units in the softmax layer was set to the number of classes in the dataset. ReLU activation was used. All filters were 3×3. Dropout [214] was used for model regularisation with rate set at 0.5 in the two fully connected layers.

6.3.4 Performance Metrics

Multiclass CNN classifiers are most often evaluated using overall accuracy [33], however this does not necessarily reflect true usefulness of a classifier on imbalanced data. Take for example, an imbalanced dataset where the majority class represents 95% of the dataset. In this scenario, an accuracy of 95% would be obtained using a classifier that predicts the majority class for all instances. Therefore, in addition to overall accuracy, class wise values for precision, recall and F1-score are presented.

6.4 Experiments and Results

Three experiments were performed to analyse data level methods for imbalanced data classification. The first experiment uses the CNN to classify the imbalanced symbol dataset. The second experiment trains the CNN with a fully balanced symbol dataset.

The third experiment uses the multiclass imbalanced data method, prior to CNN classification. A demo of the generation and comparison of performance between the original and the CDSMOTE version of the symbol dataset can be found here[2].

6.4.1 Setup

The dataset was split, using stratification, 70:30 into training and test sets respectively. No validation set was used due to the low number of instances in the underrepresented classes. Significantly underrepresented classes (i.e. that contained one instance) were excluded from the experiment. The batch size was set to 64. The CNN was trained for 10 epochs. This training setting was selected as the experiment is designed to provide a comparative analysis of the imbalance method against the baseline, as opposed to optimise classifier performance.

6.4.2 Baseline Experiment

The CNN trained on the original dataset obtained a classification accuracy of 90.3% on the test data. Of the 2305 symbols in the test set, 222 were missclassified. A weighted average precision of 0.871, recall of 0.904 and F1-score of 0.882 was reported on the symbols in the test set. The precision, recall and F1-score per class in the test set are presented in Table 6.1. It was observed that classification performance decreases as the number of instances per class in the training set decreases, as shown in Table 6.1. This finding is consistent with observations from the literature, as it was reported that classification models trained on imbalanced datasets can be biased towards the majority classes [189]. For example, the majority class, *Sensor*, contained 1991 instances in the training set. On the majority class, the CNN reported precision, recall and F1-score of 0.98, 1.00 and 0.99 respectively. In contrast, a precision, recall and F1-score of 0.00, 0.00 and 0.00, respectively was recorded for all 15 classes with fewer than 42 instances in the training set.

6.4.3 Fully Balanced Dataset

Classification performance of the CNN trained on a dataset with equal class distribution was evaluated. To equally balance the dataset, the minority class sizes were increased to the majority class size (see Figure 6.2). To obtain equal class distribution required 64,308 additional samples. This resulted in the model training time increasing by a factor of 14.5, compared to the baseline. The SMOTE algorithm [211] was used to generate additional samples. The resulting dataset is fully balanced across all classes, compared to the original dataset with ir of 332. The fm is 0, compared to 0.972 for

[2] https://github.com/carlosfmorenog/CDSMOTE-NONBIN-Symbols

Table 6.1: CNN classification performance using Baseline, SMOTE (fully balanced) and Multiclass imbalance. **Bold** shows the best precision, recall and F1-score for each symbol. Underline indicates when the best result was obtained for a single dataset.

Class	Baseline					SMOTE Fully Balanced				CDSMOTE Multiclass Imbalanced			
	No. in test	No. in train	Precision	Recall	F1-score	No. in train	Precision	Recall	F1-score	No. in train	Precision	Recall	F1-score
Sensor	854	1991	0.98	**1.00**	0.99	1991	**1.00**	**1.00**	**1.00**	1991	**1.00**	**1.00**	**1.00**
Valve	275	640	0.93	0.98	0.96	1991	**0.99**	**0.99**	**0.99**	640	0.98	0.98	0.98
Reducer	263	614	0.96	**1.00**	0.98	1991	0.97	**1.00**	0.98	614	**0.98**	**1.00**	**0.99**
Flange Joint	126	293	0.89	**0.99**	0.94	1991	**0.98**	0.98	**0.98**	293	0.93	0.95	0.94
Continuity Label	87	201	0.95	**1.00**	0.97	1991	0.99	**1.00**	**0.99**	201	**1.00**	0.95	0.98
Valve Ball Type 2	79	183	**0.98**	**1.00**	**0.99**	1991	0.93	**1.00**	0.96	183	0.96	0.97	0.97
Valve Globe Type 1	77	179	0.91	**0.97**	0.94	1991	0.99	0.96	0.97	179	**1.00**	0.96	**0.98**
Arrowhead	72	169	0.99	0.97	0.98	1991	**1.00**	**0.99**	**0.99**	169	0.93	**0.99**	0.96
Valve Ball Type 1	52	121	0.70	0.98	0.82	1991	0.91	**0.98**	0.94	121	**0.98**	**0.98**	**0.98**
DB&BBV	43	101	0.53	0.93	0.68	1991	**1.00**	**1.00**	**1.00**	126	0.98	0.91	0.94
Valve Check	38	88	0.92	0.63	0.75	1991	0.97	**1.00**	**0.99**	126	**1.00**	0.89	0.94
DB&BPV	34	79	0.54	0.56	0.55	1991	**1.00**	**1.00**	**1.00**	126	0.82	0.97	0.89
Valve Plug	26	62	0.61	0.65	0.63	1991	**0.96**	**0.92**	**0.94**	126	0.79	0.85	0.81
ESDV Valve Ball	26	62	0.37	0.88	0.52	1991	0.89	**0.92**	**0.91**	126	**0.95**	0.81	0.88
Arrowhead + Triangle	25	58	0.75	0.72	0.73	1991	**1.00**	**0.84**	**0.91**	126	**1.00**	**0.84**	**0.91**
Valve Needle Type 1	23	54	0.64	0.78	0.71	1991	**0.96**	**1.00**	**0.98**	126	0.85	**1.00**	0.92
Triangle	22	52	0.88	0.64	0.74	1991	**0.92**	**1.00**	**0.96**	126	0.86	0.82	0.84
Valve Butterfly Type 1	21	50	**1.00**	0.86	0.92	1991	**1.00**	**0.90**	**0.95**	126	**1.00**	0.86	0.92
Flange Single T-Shape	19	45	**1.00**	0.89	0.94	1991	0.90	**0.95**	0.92	126	**1.00**	**0.95**	**0.97**
Control Valve	18	42	**1.00**	0.22	0.36	1991	0.94	**0.83**	**0.88**	126	0.92	0.67	0.77
Valve Angle	15	36	0.00	0.00	0.00	1991	0.82	0.60	0.69	126	**0.83**	**0.67**	**0.74**
Injector Point	13	30	0.00	0.00	0.00	1991	**0.92**	**0.92**	**0.92**	126	**0.92**	**0.92**	**0.92**
Tie In Point	12	29	0.00	0.00	0.00	1991	**0.92**	**0.92**	**0.92**	126	0.89	0.67	0.76
Spectacle Blind	13	29	0.00	0.00	0.00	1991	**1.00**	**1.00**	**1.00**	126	0.93	**1.00**	0.96
DB&BBV + Valve Check	12	27	0.00	0.00	0.00	1991	0.92	**1.00**	0.96	126	**1.00**	**1.00**	**1.00**
Valve Needle Type 2	10	23	0.00	0.00	0.00	1991	0.73	**0.80**	0.76	126	**1.00**	0.70	**0.82**
Valve Globe Type 2	8	20	0.00	0.00	0.00	1991	**0.88**	**0.88**	**0.88**	126	0.86	0.75	0.80
Valve Slab Gate	7	17	0.00	0.00	0.00	1991	**0.71**	0.71	**0.71**	126	0.35	**0.86**	0.50
Three Way Valve	7	17	0.00	0.00	0.00	1991	0.83	**0.71**	**0.77**	126	**1.00**	0.57	0.73
Control Valve Globe	7	16	0.00	0.00	0.00	1991	**1.00**	0.71	**0.83**	126	0.50	**0.86**	0.63
Control	6	14	0.00	0.00	0.00	1991	**1.00**	**0.67**	**0.80**	126	**1.00**	**0.67**	**0.80**
Flange + Triangle	5	12	0.00	0.00	0.00	1991	**1.00**	**0.80**	**0.89**	126	0.67	**0.80**	0.73
Exit to Atmosphere	4	10	0.00	0.00	0.00	1991	**1.00**	**0.50**	**0.67**	126	**1.00**	**0.50**	**0.67**
Rupture Disc	3	7	0.00	0.00	0.00	1991	**1.00**	**0.67**	**0.80**	126	**1.00**	**0.67**	**0.80**
ESDV Valve Slab Gate	3	6	0.00	0.00	0.00	1991	**0.67**	**0.67**	**0.67**	126	**0.67**	**0.67**	**0.67**
Weighted Average	n/a	n/a	0.871	0.904	0.882	n/a	**0.977**	**0.977**	**0.976**	n/a	0.938	0.905	0.913

the original dataset.

The SMOTE generated samples resembled realistic symbols in many cases, as can be seen in Figure 6.3 and Figure 6.4. Visually acceptable synthetic symbols were generated in two scenarios, see Figure 6.3a and Figure 6.4a. In the first scenario, the SMOTE algorithm selected two visually similar minority class samples to interpolate between. In the second scenario, the selected sample and the chosen sample from the original sample's 5 nearest neighbours are not visually close in appearance. However the randomly chosen point between the two samples is selected such that a realistic instance is generated. Synthetic instances that showed more interpolation between two visually dissimilar samples were also generated, as shown in Figure 6.3b and Figure 6.4b.

The CNN trained on the fully balanced dataset obtained an overall classification accuracy of 97.7% on the test data. Of the 2,305 symbols in the test set, 54 were missclassified. A weighted average precision of 0.977, recall of 0.977 and F1-score of 0.976 was reported on the symbols in the test set. A precision, recall and F1-score of 1.00 was observed for 4 classes *Sensor*, *DB&BBV*, *DB&BPV* and *Spectacle Blind*. A precision, recall and F1-score of 0.50 or above was obtained across all classes in the dataset. Results showed slightly lower performance for the classes with lower original class sizes. This finding may result from the CNN overfitting to the training data. The lowest values of F1-score were obtained for classes where 1,981 or higher synthetic samples

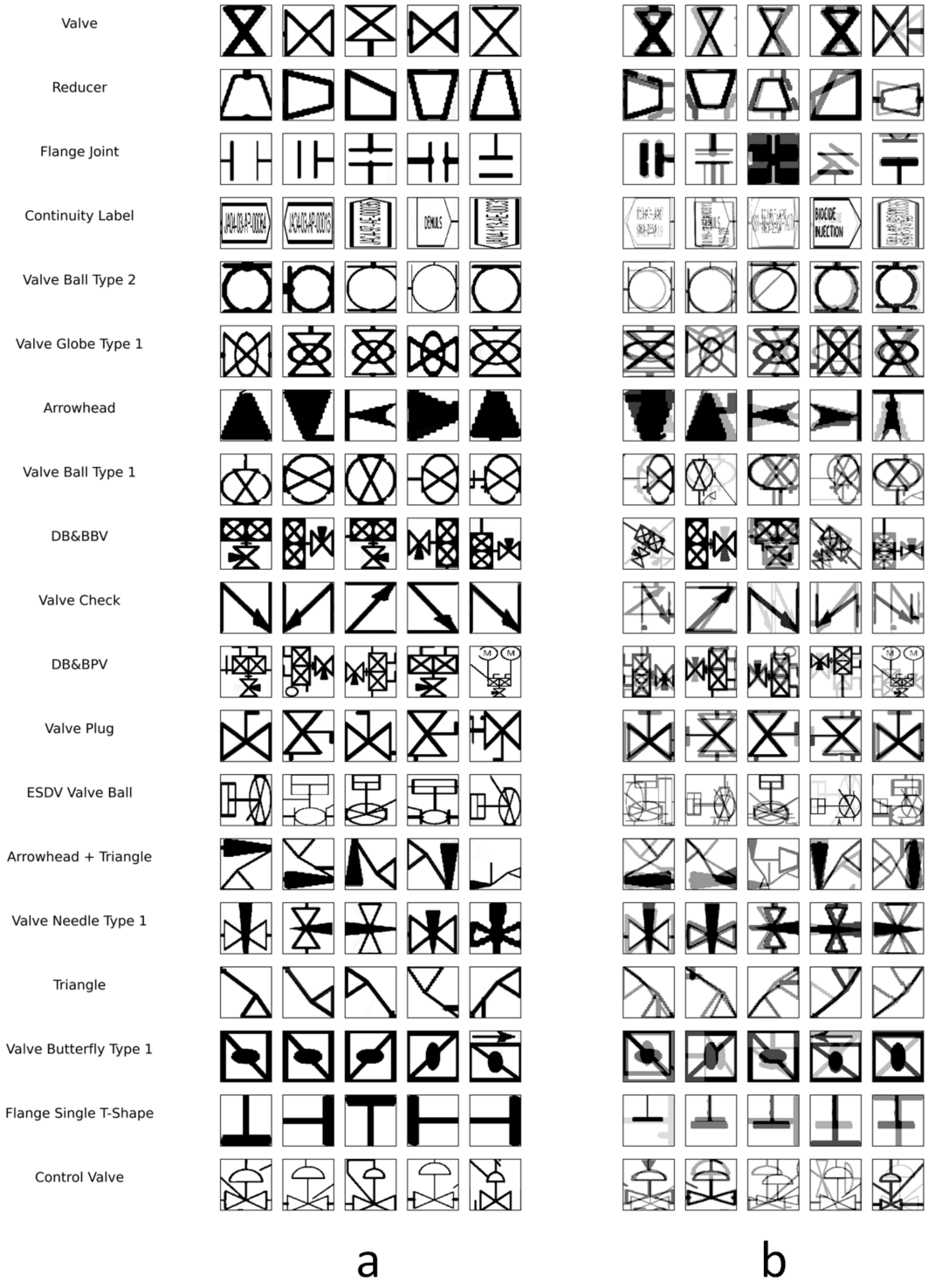


Figure 6.3: Synthetically generated minority class samples for classes Valve to Control Valve. a) realistic samples b) less realistic samples.

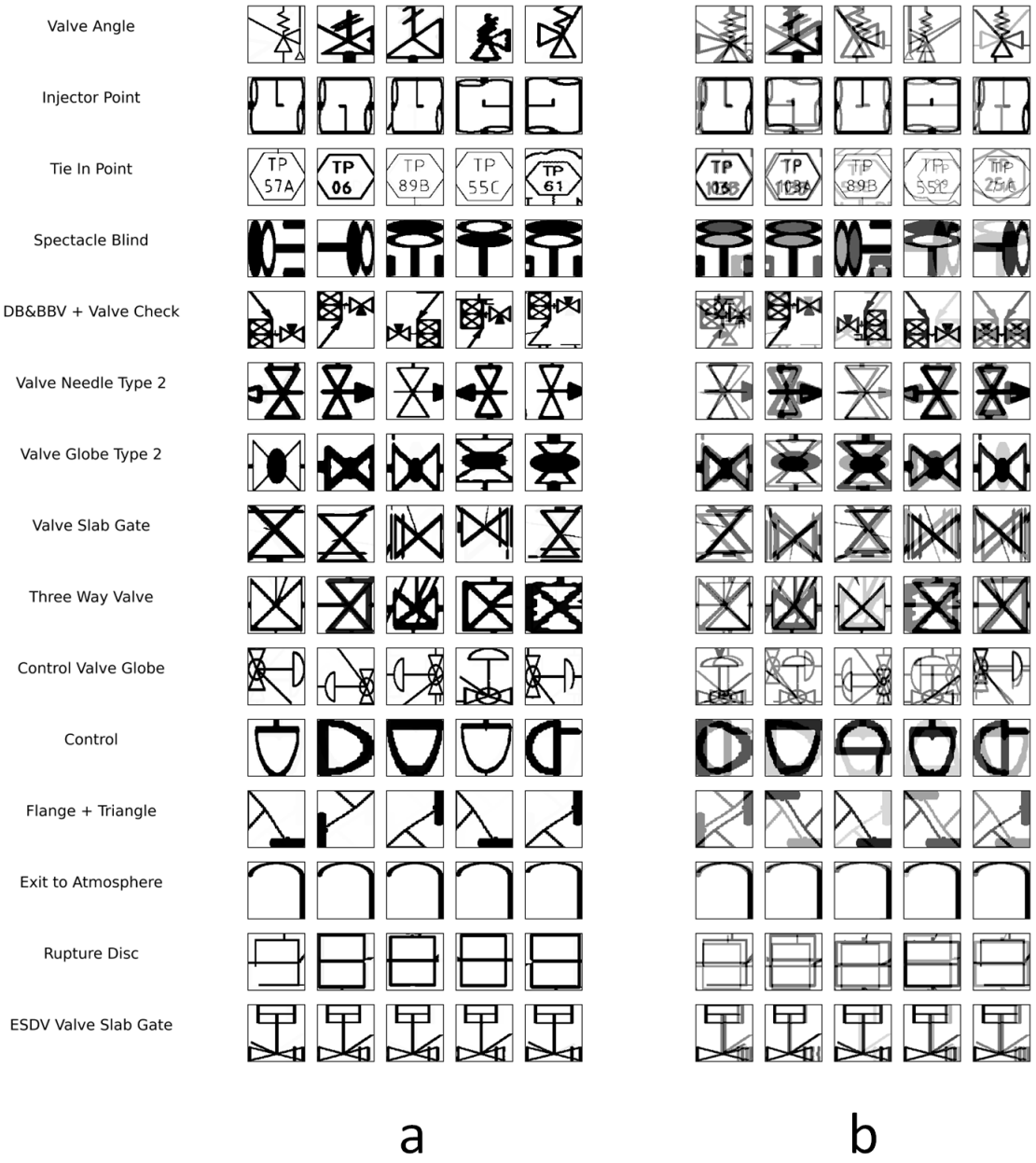


Figure 6.4: Synthetically generated minority class samples for classes Valve Angle to ESDV Valve Slab Gate. a) realistic samples b) less realistic samples.

were generated using information from 10 or fewer original samples.

6.4.4 Multiclass Imbalance Experiment

The multiclass imbalance method was used in the third experiment. The original majority class was decomposed into 10 subclasses. Synthetic oversampling of 26 minority classes occurred using SMOTE. The resulting dataset consisted of $7,667$ instances compared to $5,377$ in the original dataset. A more evenly balanced class distribution was obtained as a result of applying the multiclass imbalance data technique, as seen in Figure 6.2. The method has altered the ir of the dataset considerably from 332 to 16. The fm remains the same as the original dataset at 0.972. The CNN obtained a classification accuracy of 96.1% on the decomposed dataset, with 88 out of 2,305 symbols missclassified. A prediction of the majority class was considered correct if the prediction was of any one of the decomposed majority subclasses. A weighted average precision, recall and F1-score of 0.938, 0.905 and 0.913 respectively was reported on the decomposed dataset.

Classification performance across the minority classes has been improved as a result of applying the multiclass imbalance data technique, as can be observed in Table 6.1. Examining the results obtained for the decomposed dataset, with one exception, the precision, recall and F1-scores of 0.50 or above were observed for all classes in the dataset. Compared to the fully balanced dataset, the multiclass imbalance method obtains higher F1-score for eight classes, whereas the fully balanced dataset obtains a higher F1-score for 21 classes (Table 6.1). The reported classification performance is equal using the fully balanced training dataset compared to the multiclass imbalance method for seven classes in the dataset. These classes are *Sensor*, *Injector Point*, *Exit to Atmosphere*, *Rupture Disc*, *ESDV Valve Slab Gate*, *Control* and *Arrowhead + Triangle*.

To compare the classification results obtained with the multiclass imbalance method against baseline results, the t-test was completed. A *p-value* of 0.0000013 was obtained which shows statistically significant improvement was achieved using the proposed multiclass imbalance technique compared to the baseline method in the first experiment.

Whilst the optimum performance for certain classes was obtained using the fully balanced method, across the whole dataset the results obtained using the multiclass imbalance method are comparable whilst a much shorter training time is needed. Using the fully balanced method required a training dataset of $69,685$ instances, compared to $5,377$ with the baseline. This resulted in the model training time increasing by a factor of 14.5. In contrast, the multiclass imbalance method required only $7,667$ training instances, which increased the training time by a factor of 1.4 compared to the baseline.

6.5 Summary

In this chapter, the lack of engineering symbol datasets is addressed by presenting a labelled dataset of symbols from P&IDs and making it available in the public domain. The dataset consists of 7,728 symbols, distributed across 48 classes, from two distinct P&ID drawing standards. Engineering equipment is unevenly represented within P&IDs and as a result, the symbol dataset is significantly imbalanced. A new method for handling multiclass imbalanced classification is also described here. The method redistributes the data distribution within the dataset, whilst avoiding information loss and is an extension of CDSMOTE. Classification experiments on the symbols dataset demonstrated that the multiclass imbalanced data method improves CNN classification performance across multiple minority classes. This was obtained with a much smaller increase in training dataset size and model training time compared to the fully balanced method.

Future research will utilise the engineering symbols dataset for development and evaluation of deep learning models to digitise EDs. Additionally, further methods could be investigated to avoid biased learning models for multiclass imbalanced data, which is inherent within EDs. Generation of artificial samples to rebalance the dataset using GANs is suggested as one possible direction.

Chapter 7

Conclusion and Future Work

This chapter provides a summary of the work presented in this thesis on deep learning for digitising complex EDs. It also includes a discussion of the research limitations and suggestions for future work.

7.1 Summary

This thesis presented an end-to-end deep learning framework for the digitisation of complex EDs. It is also worthwhile pointing out that this research had real-world impact as the frameworks created were deployed in several companies, including DNV and Fieri Analytics/TaksoAI. These frameworks allowed for the automatic recognition of symbols, text and additionally pipelines within complex EDs. An example of this is in Figure 7.1. This shows a section of a processed drawing with all of the predictions overlaid. Additionally, a csv output file was created that listed all of the automatically recognised components and their corresponding details. For instance, each recognised symbol was listed with the predicted class, item number, top left x,y co-ordinate, and width and height of the bounding box. Furthermore, the implementation allowed the symbol digitisation method to be evaluated by comparing the predicted symbols to the ground truth.

All of the research presented here was evaluated using real-world datasets sourced from industry. Note that this differs from a large proportion of the existing literature which utilised simplified or synthetic EDs [43, 11, 6]. A range of drawings sourced from various sectors, specifically construction and oil and gas, was used in this thesis. Note that these drawings are high resolution images which differs from the much smaller images seen in popular computer vision datasets such as ImageNet [207]. Furthermore, the construction dataset contains drawings from various projects for different building

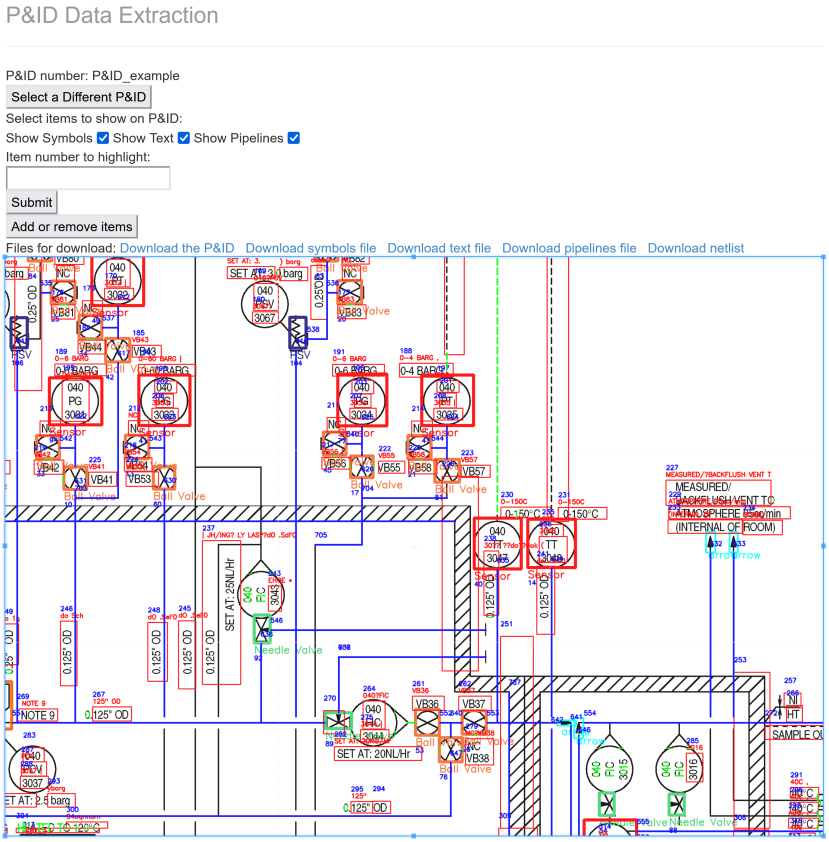


Figure 7.1: Drawing digitisation framework such as deployed in industry. Shows section of processed drawing. Note that all colour annotations represent the results.

types. It was important to use real-world datasets are they are typically more complex than synthetic drawings, for instance they usually contain a wider variety of symbols and layouts. Moreover, using drawings from industry means that the findings presented here are relevant to realistic application scenarios.

The thesis provides several contributions to the research field of deep learning for ED digitisation. The contributions meet the research objectives defined in Section 1.3 and are summarised, along with the main findings, as follows:

- An extensive critical review of the literature on deep learning for ED digitisation was provided in Chapter 2. This reviewed the existing methods and identified the latest developments in the field. This was important as there was no published survey which thoroughly discussed the relatively new application of deep learning methods for this problem. First, the literature was reviewed and categorised according to the various application domains across different industry sectors, such as oil and gas, and architectural. Next, an in-depth critical investigation of

the state-of-the-art deep learning methods for symbols recognition, text extraction and connection detection was presented. This was followed by a thorough discussion of remaining challenges in this area which were identified as dataset availability, data annotation, class imbalance, evaluation methods and contextualisation. Suggestions for future work to improve research and development in deep learning for ED digitisation were also provided here.

- The deep learning framework was used to provide a comprehensive evaluation of deep learning text digitisation methods in real-world complex EDs, in Chapter 3. This provides a thorough analysis of these methods for a real-world application. Extensive experiments on a large dataset of P&IDs sourced from industry were presented. This involved deep learning methods for text detection and text recognition. Analysis of the results showed typical scenarios where the deep learning methods performed well and it also demonstrated where improvements could be made. For instance, certain background components were detected as text. It was also shown that the digitisation of text strings located in close proximity to other drawing elements was challenging for the deep learning models.

- A novel framework for the automatic processing and analysis of engineering drawings was presented in Chapter 4. This was thoroughly tested using real-world construction drawings. These drawings have received relatively little attention compared to others, however their analysis is essential for many crucial tasks in a construction project. One of these tasks is material takeoff, where the purpose is to generate a list of the required equipment and the respective amounts. This work is believed to be the first example of these experiments using complex construction drawings from industry. Evaluation was performed on a large dataset of high-resolution construction diagrams sourced from a range of industry projects. Various symbol classes were used, which had high levels of intra-class variability and inter-class similarity. The performance and speed of a one-stage and two-stage model architecture in this scenario was also compared. Results showed that both methods performed well across the various symbol classes. Additionally, there was a significant time saving compared to manual drawing analysis. This framework enables the digital transformation of construction drawings, improving tasks such as material take-off and many others.

- The challenge of data annotation was addressed using few-shot learning for symbol detection in complex EDs in Chapter 5. This is one of the first examples of a few-shot learning approach for the digitisation of real-world EDs. This is beneficial as most deep learning methods require a large labelled dataset for all symbols, however this can be challenging to acquire due to dataset availability

and the costly annotation effort. Acquiring sufficient instances of rare classes can also be difficult. Extensive experiments were performed to validate the few-shot approach. The results showed that the method successfully detected symbols after learning from fewer than ten labelled training instances of each novel class. Furthermore, statistically significant improvement was achieved compared to various state-of-the-art detection methods.

- A new dataset of engineering symbols was presented [1] to address the lack of datasets in this domain in Chapter 6. It contains symbols sourced from two P&ID drawing standards, and in total contains $7,728$ symbols from 48 classes. The dataset is highly imbalanced, reflecting symbol distribution in real-world drawings. Furthermore, a method to handle multiclass imbalanced classification based on class decomposition was provided for this challenging dataset. The experiments clearly show that significant improvement in classification performance was obtained, without causing information loss, as occurs in other class-imbalance data sampling approaches.

7.2 Limitations and Future Work

This thesis has made multiple contributions to the relatively new research field of deep learning for ED digitisation. However, it is a challenging problem and there are limitations to the work presented here. To address these, suggestions for future research directions are provided.

In this thesis, the focus was on the digitisation of two of the most crucial drawing elements, which are the symbols and text. To fully digitise the whole drawing also requires separate methods for connection digitisation. This is a demanding task, as these connections are represented by a wide variety of shapes. For instance, in the P&ID dataset, straight lines were used to indicate pipelines and dashed lines represented electrical connections. A wider variety of connection types was seen in the construction drawings. This dataset contained subsets of plumbing and HVAC drawings. The plumbing drawings contained a wide variety of line types to represent connections, such as various dashed lines. In the HVAC drawings, ducts were used to connect pieces of equipment. The duct shapes were more similar to other symbols than lines. Furthermore, they were typically much larger and constituted a higher proportion of the drawing compared to a typical line. These types of connections could potentially be detected using an object detection model. It is worth pointing out that this would likely perform better as a separate model from one designed for symbols due to the

[1] https://github.com/carlosfmorenog/CDSMOTE-NONBIN-Symbols

different object sizes, similar to the work presented in [32] where authors used two object detection models for different scales of symbols. It should also be noted that it would be necessary to have a ground truth dataset of connectors for evaluation purposes, which would require significant annotation effort.

Another extension to this work is to provide a level of contextualisation for these drawings. This would be of benefit to establish the relationships between the symbols, connections and text. For example, this could link a symbol to the relevant text annotations. It is suggested to use graph methods to contextualise this data, such as in [44, 43]. This would extract even more valuable information from these drawings.

The performance of the text digitisation methods presented in Chapter 3 could potentially be improved by specifically training a deep learning model on the fonts used in EDs. It would be beneficial to use training samples from the intended test distribution, as this would contain text in similar scenarios, for example in dense representation, overlapping other shapes and in various orientations and locations on the drawings. Note that this requires an annotated text dataset to be created, which is likely to be extremely time consuming.

Class imbalance exists in ED datasets, as observed in the symbol detection experiments presented in Chapter 4. This problem is when a deep learning model trained on an imbalanced dataset is biased towards the majority classes [33]. It occurs in engineering datasets due to the inherent imbalance in the use of different equipment. The few-shot detection approach proposed in Chapter 5 was shown to improve performance for minority classes with few training samples. Alternative approaches could also address this problem. For example, the symbol dataset could be balanced using synthetic symbols. Note that most of the existing literature accomplished this through the use of image processing data augmentation techniques [3, 2] or specialist ED software [32], however generative deep learning methods could also be used for this purpose such as symbols created using GANs [1]. A further extension of this is to generate complete patches of EDs containing symbols, connections and text. In doing so, it is important that the synthetic images closely resemble realistic scenarios likely to appear in these drawings.

One of the limitations of using a deep learning symbol detection method as presented in Chapter 4 is the costly annotation effort required to create the labelled symbol datasets. This is time-consuming and requires subject matter experts [4, 5]. The required labelled training data can be reduced using a few-shot method as presented in Chapter 5. Another potential solution to reduce the required annotation effort is to extend the deployed drawing digitisation frameworks to incorporate active learning

algorithms. These methods assume that different samples have a different impact on model performance during training, and aim to maintain model performance whilst minimising the training set [215]. They can be used to selectively query human experts to label the most informative samples. Therefore, manual annotation is reduced as fewer labelled training instances are needed.

A related suggestion for further research is human-in-the-loop deep learning [35]. Note that this was investigated as part of the work for the industry partners. The drawing digitisation framework was developed to be interactive and allow human knowledge to be incorporated into the output. For instance, in the example shown in Figure 7.2, the deep learning model identified part of a vertical dashed line as a piece of text. The user is able to correct this by selecting the relevant button and then drawing a bounding box around the piece of text to be removed. Any user corrections such as this would then be reflected in the output csv file. The human-in-the-loop approach was beneficial here as although the deep learning models showed very promising results, incorrect predictions can occur.

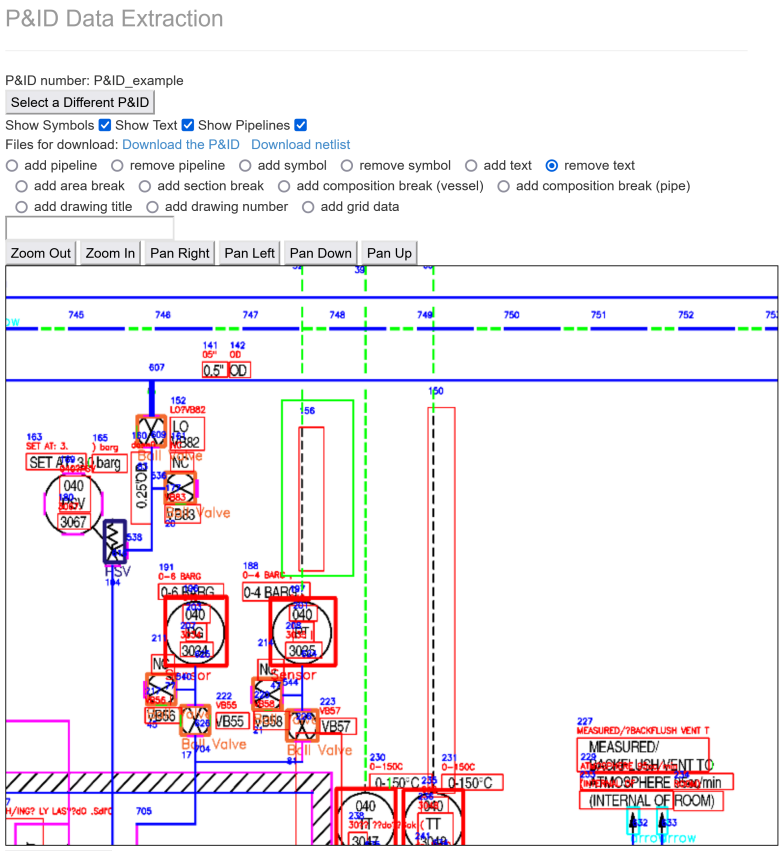


Figure 7.2: Digitisation framework enables incorporation of human knowledge in the output. Note that all colour annotations represent the results.

Additionally, the framework created allowed a human expert to review and correct model symbol predictions, as shown in Figure 7.3. False negatives can also be corrected by the addition of an undetected symbol. The manually reviewed symbol data can then be downloaded and used in an iterative training process with the aim to improve model performance. Further detailed research into approaches that combine deep learning models and selective manual input is likely to result in improved digitisation methods whilst also reducing the required annotation effort.

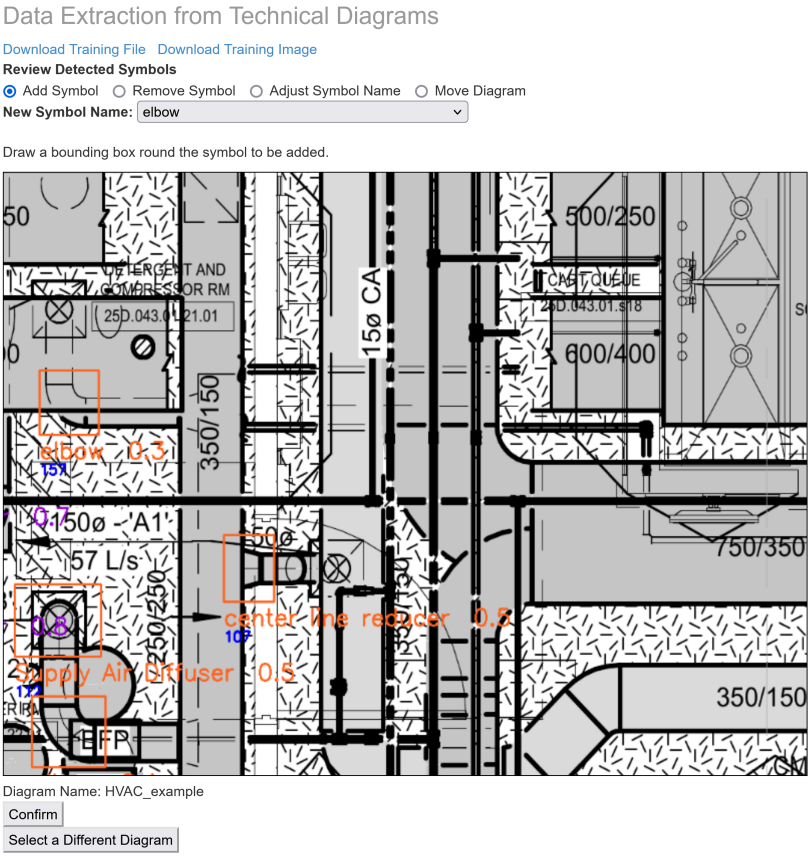


Figure 7.3: Interactive framework enables iterative model training. Note that all colour annotations represent the results.

The few-shot symbol detection method presented in Chapter 5 required fewer training instances compared to other symbol detection methods, whilst it also improved performance on the novel classes. However, it was also shown to reduce the performance on the majority classes in comparison with other methods. In order to achieve high performance across all of the classes, one potential solution is to use an ensemble approach to combine predictions from a symbol detection model and a few-shot model. Another possible solution is to balance the dataset through the use of synthetic images from generative deep learning models.

All of the research in this thesis was evaluated on complex real-world drawings. However, benchmarking of digitisation methods is difficult due to the lack of publicly available annotated datasets. For other computer vision tasks, standard datasets such as ImageNet and COCO [207, 191] are frequently used to benchmark the performance of deep learning models. Therefore, release of complex ED datasets could accelerate the research and development of digitisation methods. To provide this most benefit, these should be as realistic as possible and contain drawing elements such as those found in real-world drawing such as noise, overlapping elements and broken symbols.

Bibliography

[1] Elyan E, Jamieson L, Ali-Gombe A. Deep learning for symbols detection and classification in engineering drawings. Neural Networks. 2020;129:91-102.

[2] Jakubik J, Hemmer P, Vössing M, Blumenstiel B, Bartos A, Mohr K. Designing a Human-in-the-Loop System for Object Detection in Floor Plans. Proceedings of the AAAI Conference on Artificial Intelligence. 2022 Jun;36(11):12524-30.

[3] Gao W, Zhao Y, Smidts C. Component detection in piping and instrumentation diagrams of nuclear power plants based on neural networks. Progress in Nuclear Energy. 2020;128:103491.

[4] Joy J, Mounsef J. Automation of Material Takeoff using Computer Vision. In: 2021 IEEE International Conference on Industry 4.0, Artificial Intelligence, and Communications Technology (IAICT); 2021. p. 196-200.

[5] Liu H, Cheng JCP, Gan VJL, Zhou S. A knowledge model-based BIM framework for automatic code-compliant quantity take-off. Automation in Construction. 2022;133:104024.

[6] Paliwal S, Sharma M, Vig L. OSSR-PID: One-Shot Symbol Recognition in P amp;ID Sheets using Path Sampling and GCN. In: 2021 International Joint Conference on Neural Networks (IJCNN); 2021. p. 1-8.

[7] Khosakitchalert C, Yabuki N, Fukuda T. Automated modification of compound elements for accurate BIM-based quantity takeoff. Automation in Construction. 2020;113:103142.

[8] Kang SO, Lee EB, Baek HK. A Digitization and Conversion Tool for Imaged Drawings to Intelligent Piping and Instrumentation Diagrams P&ID. Energies. 2019;12(13).

[9] Groen FC, Sanderson AC, Schlag JF. Symbol recognition in electrical diagrams using probabilistic graph matching. Pattern Recognition Letters. 1985;3(5):343-50.

[10] Okazaki A, Kondo T, Mori K, Tsunekawa S, Kawamoto E. An automatic circuit diagram reader with loop-structure-based symbol recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1988;10(3):331-41.

[11] Nurminen JK, Rainio K, Numminen JP, Syrjänen T, Paganus N, Honkoila K. Object Detection in Design Diagrams with Machine Learning. In: Burduk R, Kurzynski M, Wozniak M, editors. Progress in Computer Recognition Systems. Cham: Springer International Publishing; 2020. p. 27-36.

[12] Ablameyko S, Uchida S. Recognition of Engineering Drawing Entities: Review of Approaches. International Journal of Image and Graphics. 2007 10;7:709-33.

[13] Moreno-García CF, Elyan E, Jayne C. New trends on digitisation of complex engineering drawings. Neural Computing and Applications. 2019;31(6):1695-712.

[14] LeCun Y, Bengio Y, Hinton G. Deep learning. nature. 2015;521(7553):436.

[15] Datta R, Mandal PDS, Chanda B. Detection and identification of logic gates from document images using mathematical morphology. In: 2015 Fifth National Conference on Computer Vision, Pattern Recognition, Image Processing and Graphics (NCVPRIPG); 2015. p. 1-4.

[16] Arroyo E, Hoernicke M, Rodríguez P, Fay A. Automatic derivation of qualitative plant simulation models from legacy piping and instrumentation diagrams. Computers & Chemical Engineering. 2016;92:112-32.

[17] LeCun Y, Bottou L, Bengio Y, Haffner P, et al. Gradient-based learning applied to document recognition. Proceedings of the IEEE. 1998;86(11):2278-324.

[18] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. In: Pereira F, Burges CJC, Bottou L, Weinberger KQ, editors. Advances in Neural Information Processing Systems 25. Curran Associates, Inc.; 2012. p. 1097-105.

[19] 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; 2016. p. 779-88.

[20] Redmon J, Farhadi A. YOLO9000: Better, Faster, Stronger. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017. p. 6517-25.

[21] Redmon J, Farhadi A. Yolov3: An incremental improvement. arXiv preprint arXiv:180402767. 2018.

[22] Bochkovskiy A, Wang CY, Liao HYM. Yolov4: Optimal speed and accuracy of object detection. arXiv preprint arXiv:200410934. 2020.

[23] Jocher G, Nishimura K, Mineeva T, Vilariño R. yolov5. Code repository https://github com/ultralytics/yolov5. 2020.

[24] Li C, Li L, Jiang H, Weng K, Geng Y, Li L, et al.. YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications. arXiv; 2022.

[25] Wang CY, Bochkovskiy A, Liao HYM. YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. arXiv; 2022.

[26] Jocher G, Chaurasia A, Qiu J. Ultralytics YOLOv8. Code repository https://githubcom/ultralytics/ultralytics. 2023.

[27] Moreno-Garcia CF, Elyan E. Digitisation of Assets from the Oil and Gas Industry: Challenges and Opportunities. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 7; 2019. p. 2-5.

[28] Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, et al. ImageNet Large Scale Visual Recognition Challenge. International journal of computer vision. 2015;115(3):211-52.

[29] Theisen MF, Flores KN, Schulze Balhorn L, Schweidtmann AM. Digitization of chemical process flow diagrams using deep convolutional neural networks. Digital Chemical Engineering. 2023;6:100072.

[30] Dzhusupova R, Banotra R, Bosch J, Olsson HH. Using artificial intelligence to find design errors in the engineering drawings. Journal of Software: Evolution and Process;n/a(n/a):e2543.

[31] Elyan E, Moreno-García CF, Johnston P. Symbols in Engineering Drawings (SiED): An Imbalanced Dataset Benchmarked by Convolutional Neural Networks. In: Iliadis L, Angelov PP, Jayne C, Pimenidis E, editors. Proceedings of the 21st EANN (Engineering Applications of Neural Networks) 2020 Conference. Cham: Springer International Publishing; 2020. p. 215-24.

[32] Kim H, Lee W, Kim M, Moon Y, Lee T, Cho M, et al. Deep-learning-based recognition of symbols and texts at an industrially applicable level from images of high-density piping and instrumentation diagrams. Expert Systems with Applications. 2021;183:115337.

[33] Buda M, Maki A, Mazurowski MA. A systematic study of the class imbalance problem in convolutional neural networks. Neural Networks. 2018;106:249-59.

[34] Rahul R, Paliwal S, Sharma M, Vig L. Automatic Information Extraction from Piping and Instrumentation Diagrams. In: Marsico MD, di Baja GS, Fred ALN, editors. Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods, ICPRAM 2019, Prague, Czech Republic, February 19-21, 2019. SciTePress; 2019. p. 163-72.

[35] Mosqueira-Rey E, Hernández-Pereira E, Alonso-Ríos D, Bobes-Bascarán J, Fernández-Leal A. Human-in-the-loop machine learning: a state of the art. Artificial Intelligence Review. 2022 aug;56(4):3005–3054.

[36] Jamieson L, Francisco Moreno-García C, Elyan E. A review of deep learning methods for digitisation of complex documents and engineering diagrams. Artificial Intelligence Review. 2024;57(6):1-37.

[37] Jamieson L, Moreno-Garcia CF, Elyan E. Deep Learning for Text Detection and Recognition in Complex Engineering Diagrams. In: 2020 International Joint Conference on Neural Networks (IJCNN); 2020. p. 1-7.

[38] Jamieson L, Moreno-Garcia CF, Elyan E. Towards fully automated processing and analysis of construction diagrams: AI-powered symbol detection. International Journal on Document Analysis and Recognition (IJDAR). 2024:1-14.

[39] Laura Jamieson EE, Moreno-García CF. Few-Shot Symbol Detection in Engineering Drawings. Applied Artificial Intelligence. 2024;38(1):2406712.

[40] Jamieson L, Moreno-García CF, Elyan E. A Multiclass Imbalanced Dataset Classification of Symbols from Piping and Instrumentation Diagrams. In: Barney Smith EH, Liwicki M, Peng L, editors. Document Analysis and Recognition - ICDAR 2024. Cham: Springer Nature Switzerland; 2024. p. 3-16.

[41] De P, Mandal S, Bhowmick P. Recognition of electrical symbols in document images using morphology and geometric analysis. In: 2011 International Conference on Image Information Processing; 2011. p. 1-6.

[42] Kim H, Kim S, Yu K. Automatic Extraction of Indoor Spatial Information from Floor Plan Image: A Patch-Based Deep Learning Methodology Application on Large-Scale Complex Buildings. ISPRS International Journal of Geo-Information. 2021;10(12).

[43] Paliwal S, Jain A, Sharma M, Vig L. Digitize-PID: Automatic Digitization of Piping and Instrumentation Diagrams. In: Gupta M, Ramakrishnan G, editors. Trends and Applications in Knowledge Discovery and Data Mining - PAKDD 2021 Workshops, WSPA, MLMEIN, SDPRA, DARAI, and AI4EPT, Delhi, India, May 11, 2021 Proceedings. vol. 12705 of Lecture Notes in Computer Science. Springer; 2021. p. 168-80.

[44] Mani S, Haddad MA, Constantini D, Douhard W, Li Q, Poirier L. Automatic Digitization of Engineering Diagrams using Deep Learning and Graph Search. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2020. p. 673-9.

[45] Hantach R, Lechuga G, Calvez P. Key Information Recognition from Piping and Instrumentation Diagrams: Where We Are? In: Barney Smith EH, Pal U, editors. Document Analysis and Recognition – ICDAR 2021 Workshops. Cham: Springer International Publishing; 2021. p. 504-8.

[46] Espina-Romero L, Guerrero-Alcedo J. Fields Touched by Digitalization: Analysis of Scientific Activity in Scopus. Sustainability. 2022;14(21).

[47] Dalal N, Triggs B. Histograms of oriented gradients for human detection. In: 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). vol. 1; 2005. p. 886-93 vol. 1.

[48] Lowe DG. Distinctive image features from scale-invariant keypoints. International journal of computer vision. 2004;60(2):91-110.

[49] Bay H, Tuytelaars T, Van Gool L. Surf: Speeded up robust features. In: European conference on computer vision. Springer; 2006. p. 404-17.

[50] Ojala T, Pietikainen M, Maenpaa T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2002 July;24(7):971-87.

[51] Yu ES, Cha JM, Lee T, Kim J, Mun D. Features Recognition from Piping and Instrumentation Diagrams in Image Format Using a Deep Learning Network. Energies. 2019;12(23).

[52] Zhao Y, Deng X, Lai H. A Deep Learning-Based Method to Detect Components from Scanned Structural Drawings for Reconstructing 3D Models. Applied Sciences. 2020;10(6).

[53] Pizarro PN, Hitschfeld N, Sipiran I, Saavedra JM. Automatic floor plan analysis and recognition. Automation in Construction. 2022;140:104348.

[54] Sinha A, Bayer J, Bukhari SS. Table Localization and Field Value Extraction in Piping and Instrumentation Diagram Images. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 1; 2019. p. 26-31.

[55] Moreno-García CF, Johnston P, Garkuwa B. Pixel-based layer segmentation of complex engineering drawings using convolutional neural networks. In: 2020 International Joint Conference on Neural Networks (IJCNN); 2020. p. 1-7.

[56] Moon Y, Lee J, Mun D, Lim S. Deep Learning-Based Method to Recognize Line Objects and Flow Arrows from Image-Format Piping and Instrumentation Diagrams for Digitization. Applied Sciences. 2021;11(21):10054.

[57] Stinner F, Wiecek M, Baranski M, Kümpel A, Müller D. Automatic digital twin data model generation of building energy systems from piping and instrumentation diagrams; 2021.

[58] Toral L, Moreno-García CF, Elyan E, Memon S. A Deep Learning Digitisation Framework to Mark up Corrosion Circuits in Piping and Instrumentation Diagrams. In: Barney Smith EH, Pal U, editors. Document Analysis and Recognition – ICDAR 2021 Workshops. Cham: Springer International Publishing; 2021. p. 268-76.

[59] Bhanbhro H, Hooi YK, Hassan Z, Sohu N. Modern Deep Learning Approaches for Symbol Detection in Complex Engineering Drawings. In: 2022 International Conference on Digital Transformation and Intelligence (ICDI); 2022. p. 121-6.

[60] Ziran Z, Marinai S. Object Detection in Floor Plan Images. In: Pancioni L, Schwenker F, Trentin E, editors. Artificial Neural Networks in Pattern Recognition. Cham: Springer International Publishing; 2018. p. 383-94.

[61] Rezvanifar A, Cote M, Albu AB. Symbol Spotting on Digital Architectural Floor Plans Using a Deep Learning-based Framework. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2020. p. 2419-28.

[62] Renton G, Balcilar M, Héroux P, Gaüzère B, Honeine P, Adam S. Symbols Detection and Classification using Graph Neural Networks. Pattern Recognition Letters. 2021;152:391-7.

[63] Nguyen T, Pham LV, Nguyen C, Nguyen VV. Object Detection and Text Recognition in Large-scale Technical Drawings. In: Proceedings of the 10th International Conference on Pattern Recognition Applications and Methods - Volume 1: ICPRAM,. INSTICC. SciTePress; 2021. p. 612-9.

[64] Faltin B, Schönfelder P, König M. Inferring Interconnections of Construction Drawings for Bridges Using Deep Learning-based Methods; 2022. p. 343-50.

[65] Francois M, Eglin V, Biou M. Text Detection and Post-OCR Correction in Engineering Documents. In: Uchida S, Barney E, Eglin V, editors. Document Analysis Systems. Cham: Springer International Publishing; 2022. p. 726-40.

[66] Renton G, Héroux P, Gaüzère B, Adam S. Graph Neural Network for Symbol Detection on Document Images. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 1; 2019. p. 62-7.

[67] Hu H, Zhang C, Liang Y. Detection of surface roughness of mechanical drawings with deep learning. Journal of Mechanical Science and Technology. 2021;35(12):5541-9.

[68] Scheibel B, Mangler J, Rinderle-Ma S. Extraction of dimension requirements from engineering drawings for supporting quality control in production processes. Computers in Industry. 2021;129:103442.

[69] Mizanur Rahman S, Bayer J, Dengel A. Graph-Based Object Detection Enhancement for Symbolic Engineering Drawings. In: Document Analysis and Recognition – ICDAR 2021 Workshops: Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part I. Berlin, Heidelberg: Springer-Verlag; 2021. p. 74–90.

[70] Bickel S, Schleich B, Wartzack S. DETECTION AND CLASSIFICATION OF SYMBOLS IN PRINCIPLE SKETCHES USING DEEP LEARNING. Proceedings of the Design Society. 2021;1:1183–1192.

[71] Sarkar S, Pandey P, Kar S. Automatic Detection and Classification of Symbols in Engineering Drawings. arXiv; 2022.

[72] Gupta M, Wei C, Czerniawski" T. "Automated Valve Detection in Piping and Instrumentation (P&ID) Diagrams". In: "Proceedings of the 39th International Symposium on Automation and Robotics in Construction, ISARC 2022". "Proceedings of the International Symposium on Automation and Robotics in Construction". "International Association for Automation and Robotics in Construction (IAARC)"; "2022". p. "630-7".

[73] Bickel S, Goetz S, Wartzack S. FROM SKETCHES TO GRAPHS: A DEEP LEARNING BASED METHOD FOR DETECTION AND CONTEXTUALISATION OF PRINCIPLE SKETCHES IN THE EARLY PHASE OF PRODUCT DEVELOPMENT. Proceedings of the Design Society. 2023;3:1975-84.

[74] Digitalization of 2D Bridge Drawings Using Deep Learning Models. In: Prof. of the 30th Int. Conference on Intelligent Computing in Engineering (EG-ICE); 2023. .

[75] Haar C, Kim H, Koberg L. AI-Based Engineering and Production Drawing Information Extraction. In: International Conference on Flexible Automation and Intelligent Manufacturing. Springer; 2023. p. 374-82.

[76] Rumalshan OR, Weerasinghe P, Shaheer M, Gunathilake P, Dayaratna E. Transfer Learning Approach for Railway Technical Map (RTM) Component Identification. In: Proceedings of Seventh International Congress on Information and Communication Technology. Springer; 2023. p. 479-88.

[77] Ali-Gombe A, Elyan E. MFC-GAN: Class-imbalanced dataset classification using Multiple Fake Class Generative Adversarial Network. Neurocomputing. 2019;361:212 221.

[78] Dai J, Li Y, He K, Sun J. R-FCN: Object Detection via Region-based Fully Convolutional Networks. CoRR. 2016;abs/1605.06409.

[79] Shi B, Bai X, Belongie S. Detecting Oriented Text in Natural Images by Linking Segments. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017. p. 3482-90.

[80] Girshick R, Donahue J, Darrell T, Malik J. Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. In: 2014 IEEE Conference on Computer Vision and Pattern Recognition; 2014. p. 580-7.

[81] Daele DV, Decleyre N, Dubois H, Meert W. An Automated Engineering Assistant: Learning Parsers for Technical Drawings. In: AAAI; 2021. .

[82] Xie L, Lu Y, Furuhata T, Yamakawa S, Zhang W, Regmi A, et al. Graph neural network-enabled manufacturing method classification from engineering drawings. Computers in Industry. 2022 11;142:103697.

[83] Ester M, Kriegel HP, Sander J, Xu X, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: kdd. vol. 96; 1996. p. 226-31.

[84] Prasad D, Gadpal A, Kapadni K, Visave M, Sultanpure K. CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2020. p. 2439-47.

[85] Vilgertshofer S, Stoitchkov D, Borrmann A, Menter A, Genc C. Recognising railway infrastructure elements in videos and drawings using neural networks. Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction. 2019;172(1):19-33.

[86] Dzhusupova R, Banotra R, Bosch J, Olsson HH. Pattern Recognition Method for Detecting Engineering Errors on Technical Drawings. In: 2022 IEEE World AI IoT Congress (AIIoT); 2022. p. 642-8.

[87] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft COCO: Common Objects in Context. In: Fleet D, Pajdla T, Schiele B, Tuytelaars T, editors. European conference on computer vision. Cham: Springer International Publishing; 2014. p. 740-55.

[88] Ren S, He K, Girshick R, Sun J. Faster R-CNN: Towards Real-time Object Detection with Region Proposal Networks. In: Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1. NIPS'15. Cambridge, MA, USA: MIT Press; 2015. p. 91-9.

[89] Zhang F, Zhai G, Li M, Liu Y. Three-branch and Mutil-scale learning for Fine-grained Image Recognition (TBMSL-Net). CoRR. 2020;abs/2003.09150.

[90] Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation. In: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2015. p. 3431-40.

[91] He K, Gkioxari G, Dollár P, Girshick R. Mask R-CNN. In: 2017 IEEE International Conference on Computer Vision (ICCV); 2017. p. 2980-8.

[92] Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM. Dynamic Graph CNN for Learning on Point Clouds. CoRR. 2018;abs/1801.07829.

[93] Girshick R. Fast R-CNN. In: 2015 IEEE International Conference on Computer Vision (ICCV); 2015. p. 1440-8.

[94] Uijlings JR, Van De Sande KE, Gevers T, Smeulders AW. Selective search for object recognition. International journal of computer vision. 2013;104(2):154-71.

[95] Everingham M, Van Gool L, Williams CK, Winn J, Zisserman A. The pascal visual object classes (voc) challenge. International journal of computer vision. 2010;88(2):303-38.

[96] He K, Zhang X, Ren S, Sun J. Deep Residual Learning for Image Recognition. In: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2016. p. 770-8.

[97] Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z. Rethinking the Inception Architecture for Computer Vision. CoRR. 2015;abs/1512.00567.

[98] Everingham M, Van Gool L, Williams CKI, Winn J, Zisserman A. The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results; 2007. http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html.

[99] Liu W, Anguelov D, Erhan D, Szegedy C, Reed SE, Fu C, et al. SSD: Single Shot MultiBox Detector. CoRR. 2015;abs/1512.02325.

[100] Cun YL, Boser B, Denker JS, Howard RE, Habbard W, Jackel LD, et al. In: Handwritten Digit Recognition with a Back-Propagation Network. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.; 1990. p. 396–404.

[101] Yun DY, Seo SK, Zahid U, Lee CJ. Deep Neural Network for Automatic Image Recognition of Engineering Diagrams. Applied Sciences. 2020;10(11).

[102] Chang J, Wang L, Meng G, Xiang S, Pan C. Deep Adaptive Image Clustering. In: 2017 IEEE International Conference on Computer Vision (ICCV); 2017. p. 5880-8.

[103] Zheng Z, Li J, Zhu L, Li H, Petzold F, Tan P. GAT-CADNet: Graph attention network for panoptic symbol spotting in CAD drawings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022. p. 11747-56.

[104] Wang CY, Yeh IH, Liao HYM. YOLOv9: Learning What You Want to Learn Using Programmable Gradient Information. arXiv; 2024.

[105] Wang A, Chen H, Liu L, Chen K, Lin Z, Han J, et al.. YOLOv10: Real-Time End-to-End Object Detection. arXiv; 2024.

[106] Lin T, Goyal P, Girshick R, He K, Dollár P. Focal Loss for Dense Object Detection. In: 2017 IEEE International Conference on Computer Vision (ICCV); 2017. p. 2999-3007.

[107] Chollet F. Xception: Deep Learning with Depthwise Separable Convolutions. In: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR); 2017. p. 1800-7.

[108] Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, et al. Generative Adversarial Nets. In: Ghahramani Z, Welling M, Cortes C, Lawrence ND, Weinberger KQ, editors. Advances in Neural Information Processing Systems 27. Curran Associates, Inc.; 2014. p. 2672-80.

[109] Viola P, Jones M. Rapid object detection using a boosted cascade of simple features. In: 2001 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). vol. 1; 2001. p. I-I.

[110] Felzenszwalb P, McAllester D, Ramanan D. A discriminatively trained, multiscale, deformable part model. In: 2008 IEEE Conference on Computer Vision and Pattern Recognition; 2008. p. 1-8.

[111] Elyan E, Garcia CM, Jayne C. Symbols Classification in Engineering Drawings. In: 2018 International Joint Conference on Neural Networks (IJCNN); 2018. p. 1-8.

[112] Ye Q, Doermann D. Text Detection and Recognition in Imagery: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2015 July;37(7):1480-500.

[113] Long S, He X, Yao C. Scene Text Detection and Recognition: The Deep Learning Era. CoRR. 2018;abs/1811.04256.

[114] Liu J, Zhong Q, Yuan Y, Su H, Du B. SemiText: Scene text detection with semi-supervised learning. Neurocomputing. 2020;407:343 353.

[115] Zhou X, Yao C, Wen H, Wang Y, Zhou S, He W, et al. EAST: An Efficient and Accurate Scene Text Detector. CoRR. 2017;abs/1704.03155.

[116] Tian Z, Huang W, He T, He P, Qiao Y. Detecting Text in Natural Image with Connectionist Text Proposal Network. CoRR. 2016;abs/1609.03605.

[117] Baek Y, Lee B, Han D, Yun S, Lee H. Character Region Awareness for Text Detection. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2019. p. 9357-66.

[118] Adams R, Bischof L. Seeded region growing. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1994;16(6):641-7.

[119] Smith R. An overview of the Tesseract OCR engine. In: 2007 9th International Conference on Document Analysis and Recognition (ICDAR). vol. 2. IEEE; 2007. p. 629-33.

[120] Chen X, Jin L, Zhu Y, Luo C, Wang T. Text Recognition in the Wild: A Survey. ACM Comput Surv. 2021 mar;54(2).

[121] Bahdanau D, Cho K, Bengio Y. Neural Machine Translation by Jointly Learning to Align and Translate. In: Bengio Y, LeCun Y, editors. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings; 2015. .

[122] Graves A, Fernández S, Gomez F, Schmidhuber J. Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks. In: Proceedings of the 23rd International Conference on Machine Learning. ICML '06. New York, NY, USA: ACM; 2006. p. 369-76.

[123] Shi B, Bai X, Yao C. An End-to-End Trainable Neural Network for Image-Based Sequence Recognition and Its Application to Scene Text Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2017 Nov;39(11):2298-304.

[124] Kiryati N, Eldar Y, Bruckstein AM. A probabilistic Hough transform. Pattern recognition. 1991;24(4):303-16.

[125] Bin OK, Hooi YK, Kadir SJA, Fujita H, Rosli LH. Enhanced Symbol Recognition based on Advanced Data Augmentation for Engineering Diagrams. International Journal of Advanced Computer Science and Applications. 2022;13(5).

[126] Sierla S, Azangoo M, Rainio K, Papakonstantinou N, Fay A, Honkamaa P, et al. Roadmap to semi-automatic generation of digital twins for brownfield process plants. Journal of Industrial Information Integration. 2021:100282.

[127] Mishra A, Alahari K, Jawahar C. Scene Text Recognition using Higher Order Language Priors. In: Proceedings of the British Machine Vision Conference. BMVA Press; 2012. p. 127.1-127.11.

[128] Jaderberg M, Simonyan K, Vedaldi A, Zisserman A. Synthetic Data and Artificial Neural Networks for Natural Scene Text Recognition. In: Workshop on Deep Learning, NIPS; 2014. .

[129] Long S, Yao C. UnrealText: Synthesizing Realistic Scene Text Images from the Unreal World. CoRR. 2020;abs/2003.10608.

[130] Veit A, Matera T, Neumann L, Matas JES, Belongie SJ. COCO-Text: Dataset and Benchmark for Text Detection and Recognition in Natural Images. ArXiv. 2016;abs/1601.07140.

[131] Karatzas D, Shafait F, Uchida S, Iwamura M, i Bigorda LG, Mestre SR, et al. ICDAR 2013 Robust Reading Competition. In: 2013 12th International Conference on Document Analysis and Recognition (ICDAR); 2013. p. 1484-93.

[132] Karatzas D, Gomez-Bigorda L, Nicolaou A, Ghosh S, Bagdanov A, Iwamura M, et al. ICDAR 2015 competition on Robust Reading. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR); 2015. p. 1156-60.

[133] Russell BC, Torralba A, Murphy KP, Freeman WT. LabelMe: a database and web-based tool for image annotation. International journal of computer vision. 2008;77(1):157-73.

[134] Khallouli W, Pamie-George R, Kovacic S, Sousa-Poza A, Canan M, Li J. Leveraging Transfer Learning and GAN Models for OCR from Engineering Documents. In: 2022 IEEE World AI IoT Congress (AIIoT); 2022. p. 015-21.

[135] Zhu JY, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: 2017 IEEE International Conference on Computer Vision (ICCV); 2017. p. 2223-32.

[136] Fogel S, Averbuch-Elor H, Cohen S, Mazor S, Litman R. ScrabbleGAN: Semi-Supervised Varying Length Handwritten Text Generation. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2020. p. 4323-32.

[137] Van Engelen JE, Hoos HH. A survey on semi-supervised learning. Machine Learning. 2020;109(2):373-440.

[138] Zhang D, Han J, Cheng G, Yang MH. Weakly Supervised Object Localization and Detection: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence. 2022;44(9):5866-85.

[139] Ch'ng CK, Chan CS. Total-text: A comprehensive dataset for scene text detection and recognition. In: 2017 14th International Conference on Document Analysis and Recognition (ICDAR). vol. 1. IEEE; 2017. p. 935-42.

[140] Gupta A, Vedaldi A, Zisserman A. Synthetic Data for Text Localisation in Natural Images. In: IEEE Conference on Computer Vision and Pattern Recognition; 2016. .

[141] Baek Y, Nam D, Park S, Lee J, Shin S, Baek J, et al. CLEval: Character-Level Evaluation for Text Detection and Recognition Tasks. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2020. p. 2404-12.

[142] Johnson JM, Khoshgoftaar TM. Survey on deep learning with class imbalance. Journal of Big Data. 2019;6(1):27.

[143] Howie C, Kunz J, Binford T, Chen T, Law KH. Computer interpretation of process and instrumentation drawings. Advances in Engineering Software. 1998;29(7):563 570.

[144] Wen R, Tang W, Su Z. Topology based 2D engineering drawing and 3D model matching for process plant. Graphical Models. 2017;92:1-15.

[145] Wen R, Tang W, Su Z. Measuring 3D process plant model similarity based on topological relationship distribution. Computer-Aided Design and Applications. 2017;14(4):422-35.

[146] Rantala M, Niemistö H, Karhela T, Sierla S, Vyatkin V. Applying graph matching techniques to enhance reuse of plant design information. Computers in Industry. 2019;107:81-98.

[147] Sierla S, Azangoo M, Fay A, Vyatkin V, Papakonstantinou N. Integrating 2D and 3D Digital Plant Information Towards Automatic Generation of Digital Twins. In: 2020 IEEE 29th International Symposium on Industrial Electronics (ISIE); 2020. p. 460-7.

[148] Rica E, Moreno-García CF, Álvarez S, Serratosa F. Reducing human effort in engineering drawing validation. Computers in Industry. 2020;117:103198.

[149] Rica E, Álvarez S, Serratosa F. Group of components detection in engineering drawings based on graph matching. Engineering Applications of Artificial Intelligence. 2021;104:104404.

[150] Antonelli S, Avola D, Cinque L, Crisostomi D, Foresti GL, Galasso F, et al. Few-Shot Object Detection: A Survey. ACM Comput Surv. 2022 feb. Just Accepted.

[151] Tombre K, Tabbone S, Lamiroy B, Dosch P. Text/Graphics Separation Revisited. In: Document Analysis Systems. vol. 2423; 2002. p. 200-11.

[152] Dori D, Velkovitch Y. Segmentation and Recognition of Dimensioning Text from Engineering Drawings. Computer Vision and Image Understanding. 1998;69(2):196-201.

[153] Cao R, Tan CL. Text/Graphics Separation in Maps. In: 2002 3rd International Conference on Document Analysis and Recognition (ICDAR); 2002. p. 167-77.

[154] Gao F, Huang T, Sun J, Wang J, Hussain A, Yang E. A new algorithm for SAR image target recognition based on an improved deep convolutional neural network. Cognitive Computation. 2019;11(6):809-24.

[155] Ieracitano C, Adeel A, Morabito FC, Hussain A. A novel statistical analysis and autoencoder driven intelligent intrusion detection approach. Neurocomputing. 2020;387:51 62.

[156] Wang H, Zhang Z. Text Detection Algorithm based on Improved YOLOv3. In: 2019 IEEE 9th International Conference on Electronics Information and Emergency Communication (ICEIEC); 2019. p. 147-50.

[157] Liao M, Shi B, Bai X, Wang X, Liu W. TextBoxes: A Fast Text Detector with a Single Deep Neural Network. CoRR. 2016;abs/1611.06779.

[158] Liao M, Shi B, Bai X. TextBoxes++: A Single-Shot Oriented Scene Text Detector. CoRR. 2018;abs/1801.02765.

[159] Busta M, Neumann L, Matas J. Fastext: Efficient unconstrained scene text detector. In: 2015 IEEE International Conference on Computer Vision (ICCV); 2015. p. 1206-14.

[160] Kulkarni CR, Barbadekar AB. Text Detection and Recognition: A Review. International Research Journal of Engineering and Technology (IRJET). 2017;4(6):179-85.

[161] Ravagli J, Ziran Z, Marinai S. Text Recognition and Classification in Floor Plan Images. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 1; 2019. p. 1-6.

[162] Fletcher LA, Kasturi R. A robust algorithm for text string separation from mixed text/graphics images. IEEE Transactions on Pattern Analysis and Machine Intelligence. 1988;10(6):910-8.

[163] Lu T, Tai CL, Su F, Cai S. A new recognition model for electronic architectural drawings. Computer-Aided Design. 2005;37(10):1053-69.

[164] Tan CL, Ng PO. Text extraction using pyramid. Pattern Recognition. 1998;31(1):63-72.

[165] Moreno-García CF, Elyan E, Jayne C. Heuristics-Based Detection to Improve Text / Graphics Segmentation in Complex Engineering Drawings. In: Engineering Applications of Neural Networks. vol. CCIS 744; 2017. p. 87-98.

[166] Das S, Banerjee P, Seraogi B, Majumder H, Mukkamala S, Roy R, et al. Hand-Written and Machine-Printed Text Classification in Architecture, Engineering Construction Documents. In: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR); 2018. p. 546-51.

[167] Cao R, Tan CL. Separation of Touching Text from Graphics. In: Graphics Recognition Methods and Applications (GREC); 2001. p. 5-8.

[168] Brock A, Lim T, Ritchie J, Weston N. ConvNet-Based Optical Recognition for Engineering Drawings. In: International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. vol. 01; 2017. .

[169] Sermanet P, Eigen D, Zhang X, Mathieu M, Fergus R, LeCun Y. OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks. CoRR. 2013;abs/1312.6229.

[170] Lin T, Dollár P, Girshick RB, He K, Hariharan B, Belongie SJ. Feature Pyramid Networks for Object Detection. CoRR. 2016;abs/1612.03144.

[171] Eman E, Bukhari SS, Jenckel M, Dengel A. Cursive Script Textline Image Transformation for Improving OCR Accuracy. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 5; 2019. p. 59-64.

[172] Karatzas D, Gomez-Bigorda L, Nicolaou A, Ghosh S, Bagdanov A, Iwamura M, et al. ICDAR 2015 competition on robust reading. In: 2015 13th International Conference on Document Analysis and Recognition (ICDAR). IEEE; 2015. p. 1156-60.

[173] Yao C, Bai X, Liu W, Ma Y, Tu Z. Detecting texts of arbitrary orientations in natural images. In: 2012 IEEE Conference on Computer Vision and Pattern Recognition; 2012. p. 1083-90.

[174] Jaderberg M, Simonyan K, Vedaldi A, Zisserman A. Reading Text in the Wild with Convolutional Neural Networks. International journal of computer vision. 2016 jan;116(1):1-20.

[175] Zhang Z, Zhang C, Shen W, Yao C, Liu W, Bai X. Multi-oriented text detection with fully convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition; 2016. p. 4159-67.

[176] Huang L, Yang Y, Deng Y, Yu Y. DenseBox: Unifying Landmark Localization with End to End Object Detection; 2015.

[177] Kingma DP, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:14126980. 2014.

[178] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Comput. 1997 Nov;9(8):1735–1780.

[179] Monteiro A, Poças Martins J. A survey on modeling guidelines for quantity takeoff-oriented BIM-based design. Automation in Construction. 2013;35:238-53.

[180] Chowdhury AM, Moon S. Generating integrated bill of materials using mask R-CNN artificial intelligence model. Automation in Construction. 2023;145:104644.

[181] Baduge SK, Thilakarathna S, Perera JS, Arashpour M, Sharafi P, Teodosio B, et al. Artificial intelligence and smart vision for building and construction 4.0: Machine and deep learning methods and applications. Automation in Construction. 2022;141:104440.

[182] Pan Y, Zhang L. Roles of artificial intelligence in construction engineering and management: A critical review and future trends. Automation in Construction. 2021;122:103517.

[183] Abioye SO, Oyedele LO, Akanbi L, Ajayi A, Davila Delgado JM, Bilal M, et al. Artificial intelligence in the construction industry: A review of present status, opportunities and future challenges. Journal of Building Engineering. 2021;44:103299.

[184] Ressel A, Schmidt-Vollus R. Reverse engineering in process automation. In: 2021 26th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA); 2021. p. 1-4.

[185] Yin M, Tang L, Zhou T, Wen Y, Xu R, Deng W. Automatic layer classification method-based elevation recognition in architectural drawings for reconstruction of 3D BIM models. Automation in Construction. 2020;113:103082.

[186] Rezvanifar A, Cote M, Albu AB. Symbol Spotting on Digital Architectural Floor Plans Using a Deep Learning-Based Framework. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW); 2020. .

[187] Khosakitchalert C, Yabuki N, Fukuda T. Improving the accuracy of BIM-based quantity takeoff for compound elements. Automation in Construction. 2019;106:102891.

[188] Wu X, Sahoo D, Hoi SCH. Recent advances in deep learning for object detection. Neurocomputing. 2020;396:39-64.

[189] Elyan E, Moreno-Garcia CF, Jayne C. CDSMOTE: class decomposition and synthetic minority class oversampling technique for imbalanced-data classification. Neural Computing and Applications. 2020 Jul.

[190] Liu Y, Sun P, Wergeles N, Shang Y. A survey and performance evaluation of deep learning methods for small object detection. Expert Systems with Applications. 2021;172:114602.

[191] Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D, et al. Microsoft coco: Common objects in context. In: European conference on computer vision. Springer; 2014. p. 740-55.

[192] Padilla R, Passos WL, Dias TLB, Netto SL, da Silva EAB. A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. Electronics. 2021;10(3).

[193] Everingham M, Eslami SA, Van Gool L, Williams CK, Winn J, Zisserman A. The pascal visual object classes challenge: A retrospective. International journal of computer vision. 2015;111:98-136.

[194] Ruzicka V, Franchetti F. Fast and accurate object detection in high resolution 4K and 8K video using GPUs. In: 2018 IEEE High Performance extreme Computing Conference (HPEC). IEEE; 2018. p. 1-7.

[195] Weiss K, Khoshgoftaar TM, Wang D. A survey of transfer learning. Journal of Big data. 2016;3(1):1-40.

[196] Wu Y, Kirillov A, Massa F, Lo WY, Girshick R. Detectron2; 2019. https://github.com/facebookresearch/detectron2.

[197] Schlagenhauf T, Netzer M, Hillinger J. Text Detection on Technical Drawings for the Digitization of Brown-field Processes. Procedia CIRP. 2023;118:372-7. 16th CIRP Conference on Intelligent Computation in Manufacturing Engineering.

[198] Liu T, Zhang L, Wang Y, Guan J, Fu Y, Zhao J, et al. Recent Few-Shot Object Detection Algorithms: A Survey with Performance Comparison. ACM Trans Intell Syst Technol. 2023 jun;14(4).

[199] Wang X, Huang TE, Darrell T, Gonzalez JE, Yu F. Frustratingly Simple Few-Shot Object Detection. In: Proceedings of the 37th International Conference on Machine Learning. ICML'20. JMLR.org; 2020. .

[200] Fan Z, Ma Y, Li Z, Sun J. Generalized few-shot object detection without forgetting. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2021. p. 4527-36.

[201] Köhler M, Eisenbach M, Gross HM. Few-Shot Object Detection: A Comprehensive Survey. IEEE Transactions on Neural Networks and Learning Systems. 2023:1-21.

[202] Kang B, Liu Z, Wang X, Yu F, Feng J, Darrell T. Few-shot object detection via feature reweighting. In: 2019 IEEE/CVF International Conference on Computer Vision (ICCV); 2019. p. 8420-9.

[203] Hou S, Liu W, Karim A, Jia Z, Jia W, Zheng Y. Few-shot Logo Detection. IET Computer Vision. 2023 may;17(5):586–598.

[204] Kaul P, Xie W, Zisserman A. Label, Verify, Correct: A Simple Few Shot Object Detection Method. In: 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR); 2022. p. 14237-47.

[205] Cheng G, Yan B, Shi P, Li K, Yao X, Guo L, et al. Prototype-CNN for few-shot object detection in remote sensing images. IEEE Transactions on Geoscience and Remote Sensing. 2021;60:1-10.

[206] Goyal P, Dollár P, Girshick R, Noordhuis P, Wesolowski L, Kyrola A, et al. Accurate, large minibatch sgd: Training imagenet in 1 hour. arXiv preprint arXiv:170602677. 2017.

[207] Deng J, Dong W, Socher R, Li L, Li K, Fei-Fei L. ImageNet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition; 2009. p. 248-55.

[208] Vuttipittayamongkol P, Elyan E. Neighbourhood-based undersampling approach for handling imbalanced and overlapped data. Information Sciences. 2020;509:47 70.

[209] Wang S, Liu W, Wu J, Cao L, Meng Q, Kennedy PJ. Training deep neural networks on imbalanced data sets. In: 2016 International Joint Conference on Neural Networks (IJCNN); 2016. p. 4368-74.

[210] Wang S, Yao X. Multiclass Imbalance Problems: Analysis and Potential Solutions. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics). 2012;42(4):1119-30.

[211] Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: Synthetic Minority over-Sampling Technique. J Artif Int Res. 2002 Jun;16(1):321–357.

[212] Fajardo VA, Findlay D, Jaiswal C, Yin X, Houmanfar R, Xie H, et al. On oversampling imbalanced data with deep conditional generative models. Expert Systems with Applications. 2021;169:114463.

[213] Krizhevsky A, Sutskever I, Hinton GE. ImageNet Classification with Deep Convolutional Neural Networks. Commun ACM. 2017 May;60(6):84-90.

[214] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research. 2014;15:1929-58.

[215] Ren P, Xiao Y, Chang X, Huang PY, Li Z, Gupta BB, et al. A Survey of Deep Active Learning. ACM Comput Surv. 2021 oct;54(9).