AI-powered text extraction from engineering drawings

University of Turku
Department of Computing
Master of Science (Tech) Thesis
Software Engineering
September 2025
Iiro Partanen

UNIVERSITY OF TURKU
Department of Computing

Iiro Partanen: AI-powered text extraction from engineering drawings

Master of Science (Tech) Thesis, 66 p., 4 app. p.
Software Engineering
September 2025

Engineering systems contain vast amounts of data in the form of engineering documents, which contain textual data about the systems they represent. In cases where engineering documents are transformed from one format to another, this textual data might be lost. For a system such as eShare that utilizes the textual data from engineering drawings to create links between the model and documents, losing this textual data causes issues.
This thesis investigates the previous implementations that are being used for extracting textual content from engineering drawings via a literature review. Each proposed implementation is benchmarked with a test set that mimics eShare's use-case. This dataset contains various types of engineering documents in differing sizes and formats. Results from the literature review are used for a case-study in which a plan is proposed for eShare to create an OCR solution that will extract textual content from engineering drawings that do not have it included in the metadata.
The literature review was able to conclude that the latest solution from PaddleOCR, PP-OCRv5 is the optimal solution to be used for eShare. However, there were limitations found with the ready-to-use pipeline, which was to be addressed. The proposed pipeline utilizes the detection and recognition model from PP-OCRv5 but implements custom intermediate processing to handle word-level detection, which is required by eShare. The proposed solution was able to extend F1 accuracy with 8% to reach an accuracy of 89% compared to PP-OCRv5's 81%.

Keywords: Optical character recognition, artificial intelligence, engineering documents, text recognition

Contents

1 Introduction

1.1 Motivation

Engineering drawings have been used for the primary information about structures and parts for a long time. Modern CAD solutions provide files that contain information of the models as metadata. However, in some cases, such as when older hand-drawn engineering drawings are scanned or when files have been converted from one format to another, some information might be lost. In such cases the files do not inhabit all necessary metadata information about the contents of the drawing. eShare is a digital twin platform that has a 3D model of an existing structure and the engineering drawings that are based on the 3D model. In cases where the engineering drawings are made with modern CAD software, eShare can link the part identification from the engineering drawings to the 3D model. A problem occurs when the files do not inhabit metadata and eShare cannot link information from the files to the 3D model. In such cases, the user must manually extract and link the information, which is tedious and time consuming. The linking could be achieved by extracting the part numbers from the documents automatically. In addition to being able to link parts between the model and documents, eShare can not provide a search for documents without metadata, which makes all the textual information from the documents valuable.

This problem of not being able to extract information from files in cases where it

is not included in the metadata is why this subject was chosen for this thesis. The data that would be available from the automatic detection of text could in turn be used for the linking and improving the reliability of the system. The gathered data and information about possible solutions that could be utilized can be implemented in future applications, which is why this research is important in a wider scope.

1.2 Research questions

The following research questions were selected:

RQ1: What are the current solutions for detecting part numbers from engineering drawings and how do they compare?

RQ2: What is the best implementation for eShare and how can this be implemented?

The first research question will be answered through a literature review in which previous papers are examined, and potential solutions are presented. These solutions are then benchmarked with a dataset that tests their applicability to eShare's use-case. The second question will be answered through a case-study in which a solution is presented that will fit eShare's unique requirements and it will be benchmarked against the results achieved from the first research question.

1.3 Structure

The structure of this thesis consists of seven chapters. This chapter being the first, where the reader is introduced to the topic, problems that are being solved, and structure of the thesis. The second chapter provides the background information so that the reader can understand all the necessary concepts and theory that will be discussed in the rest of the thesis. It starts with information about engineering

drawings, what they are, what information they contain, and what information is being extracted from them. After engineering drawings, computer vision and optical character recognition is explained. There, theory and technologies that are being used for detecting textual information from images are explained. In addition to the second chapter, also the third chapter provides background information for the reader. In the third chapter eShare is introduced by explaining the premise, functionality, and problems that are being solved with this thesis. The current solution for the problem is also presented and the issues, that the solution has, are discussed.

The fourth chapter begins the answer for the RQ1. It explains the methodologies utilized for the literature review and presents the findings of that literature review. It contains four papers, each of which proposes a solution that is like the one that is being discussed in this thesis. In the fourth chapter the first part of the RQ1 is answered. The fifth chapter will answer the second part of the RQ1. In the fifth chapter the previously proposed solutions, and other solutions that were proposed, are tested against each other. It begins by explaining the test scenario and then by presenting and analysing the results that the tests provided. This chapter also answers the RQ2 by processing the results from previous solutions and providing a solution that would best fit eShare's use-case. The proposed solution is then tested against the previous solutions, and the results are presented and discussed.

Finally, the sixth Chapter will be discussing about the thesis as a whole and provide successes and failures that were encountered during the thesis. The Chapter will also propose future research topics for this area of research. The seventh Chapter will conclude the paper with a summarization to the whole thesis.

2 Background

2.1 Engineering drawings

An engineering drawing is a representation of an object that conveys information about it. An example of an engineering drawing is visible in Figure 2.1. The image contains an engineering drawing that represents an air duct. It is represented from different views. The center view is an overall 3D view that represents the whole model. Around this overview there are 2D representations from top, side, front, and as a cut. Each 2D model has measurements and other values about the object. Engineering drawings are widely used across different engineering domains such as oil and gas, mechanical, and electrical. This wide range of fields results in different kinds of engineering drawings. Examples of engineering drawings include mechanical drawings, which convey information of mechanical parts and process and instrumentation diagrams (P&IDs), that display information of process equipment in conjunction with other devices. [1] Previously, engineering drawings were drawn by hand on paper, but modern solutions use computer aided design (CAD), which automatically translates the created objects into engineering drawings. In these cases, the CAD software translates the objects into vector images that include metadata about the actual model. [2] To bring legacy documents up to date they need to be digitalized and the information from the images must be extracted by other methods. This process of extracting information from legacy engineering

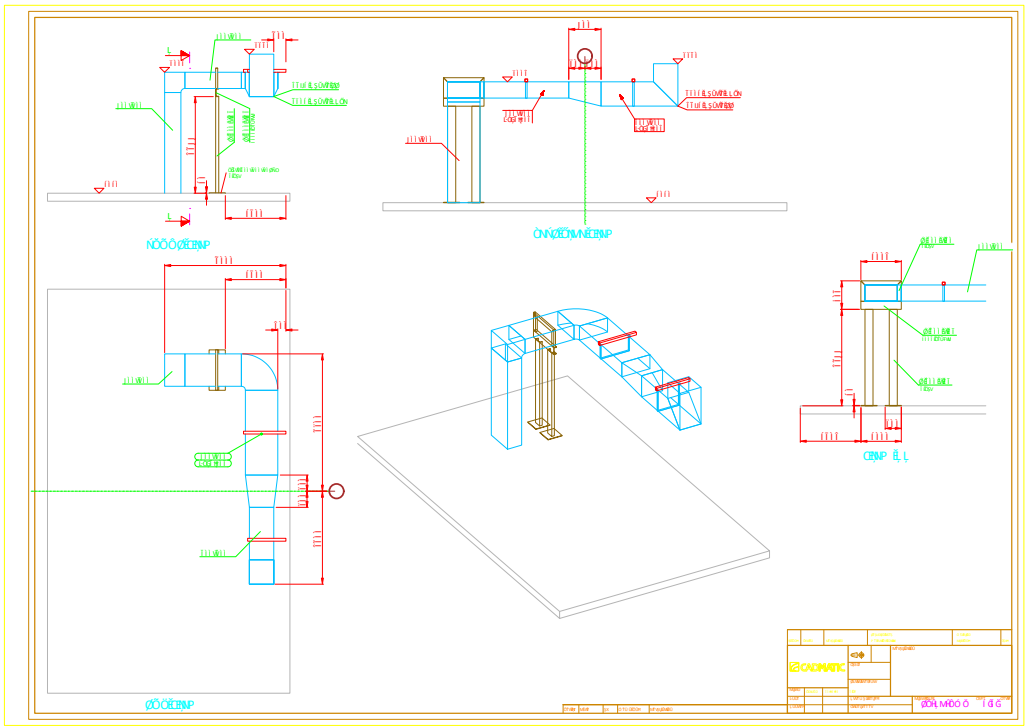

Figure 2.1: Example of an engineering drawing that represents an air duct

drawings is the core goal of this thesis.

Engineering drawings feature a wide variety of symbols that indicate different features. However, in this thesis the focus is on textual information that can be found from the engineering drawings, which means that symbols will be glossed over. Nowadays there are universal design guidelines and rules for the design of engineering drawings. These guidelines cover the type of symbols used for each feature, the layout of engineering drawings that is to be expected, and more. An example of these guidelines is ISO 128 that defines the general rules for engineering drawings [3].

2.1.1 Part numbers

There exists a lot of information in engineering drawings and the most typical information that you will find contains the title block and the model itself. The title block is not mandatory but is often found from the bottom right corner of the drawing.

It contains information about the object in textual form. Data that is considered mandatory for the title block include the drawing number, title, organization name, scale, specifications, and the author, but the title block can additionally include other information. An example of a title block is visible in Figure 2.2. In addition to the title block engineering drawings can include other tables such as the change or revision table, bill of materials, and notes. The change or revision table indicates changes made to the model, bill of material contains data about the materials used in the object, and note table is a place for notes to be made about the drawing. [4]

The model itself conveys all the information about the object that is being detailed. The model is displayed through a combination of lines from a specified view. The view can be from different perspectives such as top down, side view, and other 2D or 3D perspective. The parts themselves include information about parts dimensions, tolerances, and other quality information. [5] Singular drawings of a model can include multiple parts, which are indicated by part numbers that are displayed with textual information in the model. This is the key area of the engineering document that this thesis is trying to extract from the drawing to index it to other drawings and the part in the 3D model. An example of a multi-part drawing where the part numbers are made larger so that they are visible is in Figure 2.3. In addition to part numbers, all textual information is valuable to enable searching from the document, which makes all tables and other sources of information valuable.

2.1.2 Legacy documents

Legacy documents refer to older engineering drawings that were made prior to CAD software or with CAD, but do not have digital versions. As CAD software includes metadata to the exported engineering drawing files there is no need to extract the part numbers. In the case of legacy documents, the engineering drawings are drawn on paper and then scanned to digital formats. In these cases, the documents do not

CADMATIC SOFTWARE SOLUTIONS	Project	Contract no.
Designed by DES 16.04.18	Title Equipment & Piping Layout Unit 200	
Drawn by DRA 16.04.18		
Checked by CHE		
Approved by APR 28.04.18	Drawing no. GA-200-001	Sheet 1 of 1 Scale
		Revision Z

Figure 2.2: Example of a title block from an engineering drawing

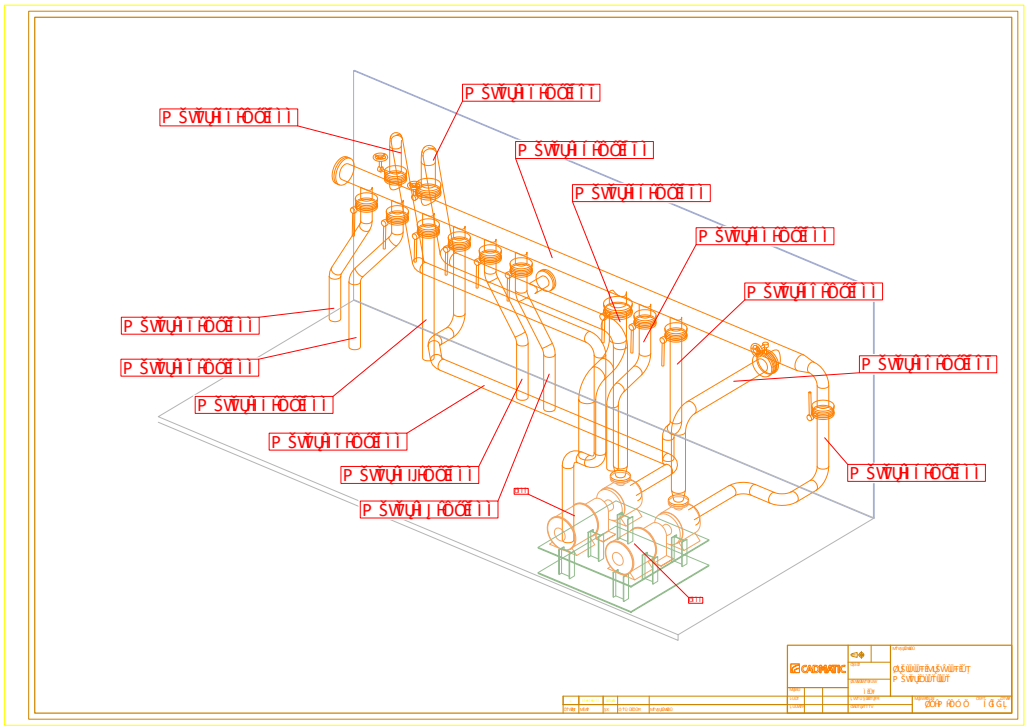

Figure 2.3: Example of a multi-part engineering drawing

have any metadata about the textual information that it might include. Sometimes older engineering drawings are scanned with proper scanners and sometimes they refer to just images taken with a camera. Legacy engineering drawings rarely include international guidelines that they follow and as such they vary more than modern engineering drawings, which results in the possibility of part numbers being present in unexpected areas. In addition to legacy documents, sometimes drawings can be made with CAD software and then exported to different formats where it might have lost information from the metadata in which case the required part numbers have to be extracted by other means.

Digitalized engineering documents might vary with the contents, but also with the scanning methodologies used. Scanning is not an easy process and often results in sub-par result. The scanning might be warped, skewed, or otherwise in bad shape. Older scanning quality was also noticeably worse than modern solutions. These are all factors that hinder the performance of text recognition from legacy engineering drawings, and they need to be considered when implementing a solution. An example of a legacy engineering drawing is visible in Figure 2.4. When compared to the modern engineering drawing in Figure 2.1 it can be seen that the legacy drawing does not have a clear structure as the side and bottom contain full width tables, there are tables without lines in the drawing area, and there are no clear sections for the models. The drawing is also skewed to the left and has varying colors, instead of uniform colors of modern drawings, which indicate that it is a low-quality scanning. Modern engineering drawings try to also avoid textual content in non-horizontal or non-vertical angles, but this legacy drawing has text orientated in multiple orientations.

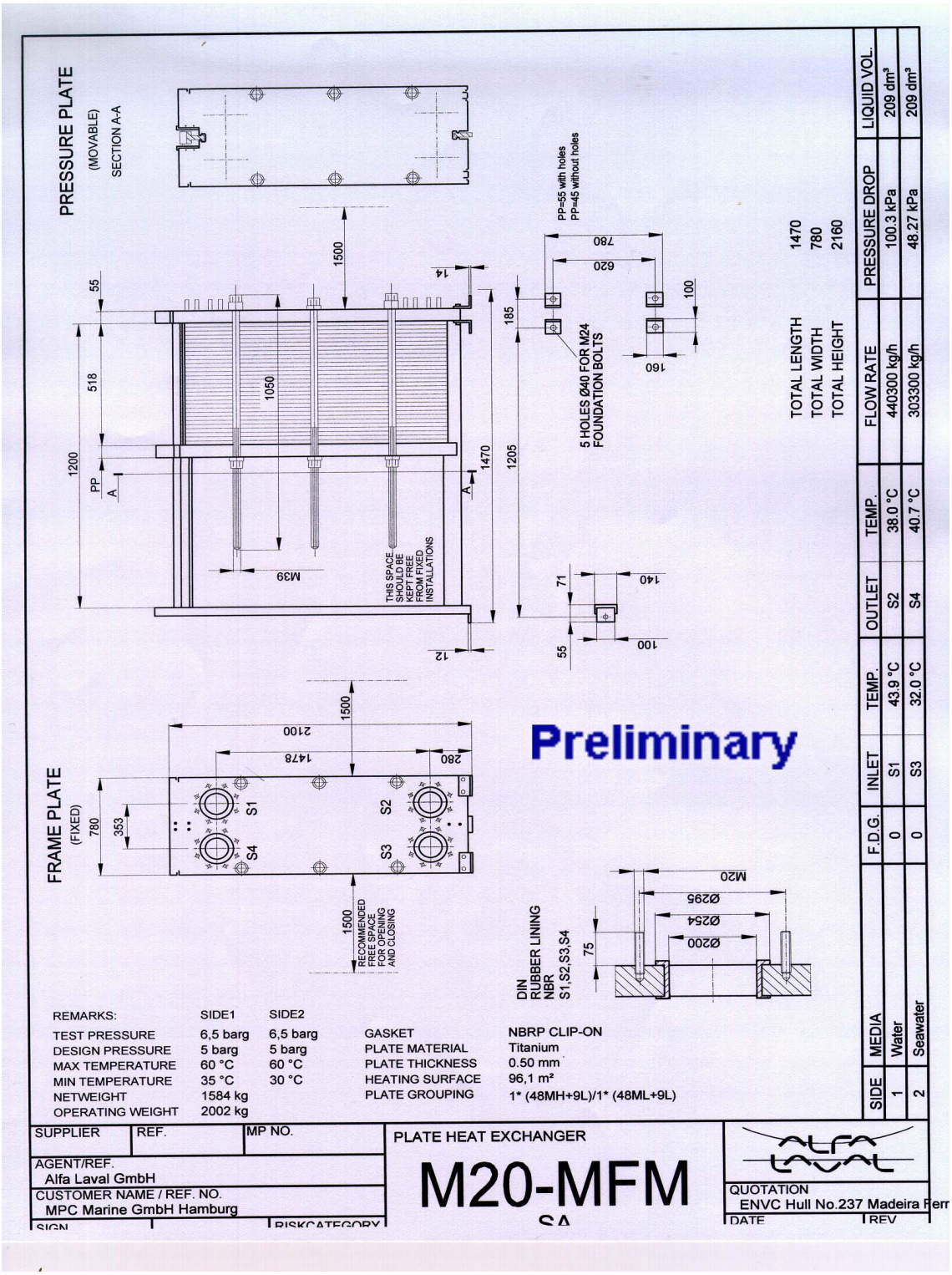

Figure 2.4: Legacy engineering drawing

2.2 Computer vision

Computer vision is a branch of Artificial Intelligence (AI) that focuses on teaching computers to understand information from visual inputs, such as images, videos, and documents. Computer vision utilizes machine learning and neural networks to understand and act on information that has been derived from the visual input. Computer vision tasks vary heavily and have use-cases from recognizing pedestrians in autonomous vehicles to detecting significant shots in golf tournaments [6]. Figure 2.5 shows the result of a computer vision model that has been tasked with detecting all the pedestrians from the input, a form of object detection. It marks the found elements with coordinates that are visualized in the figure as green rectangles. A use-case for this type of detection could be to detect pedestrians in an autonomous vehicle so that they can be taken into consideration when the model makes decisions. This thesis focuses on the tasks of object detection and text extraction in which the goal is to detect the location and content of textual data. [6], [7] Most computer vision tasks are laboursome, prone to errors, or even impossible to do by hand. Models often take in large amounts of data and are more effective and faster at handling them than humans. [7]

Computer vision requires several different technologies to work. From these the most prominent one is convolutional neural networks (CNNs) [9]. CNNs are built from convolutional, pooling, and fully connected layers. In the convolutional layer a filter with a predetermined size shifts through the images pixels and calculates a sum for each iteration with the help of an activation function and weights that are included in the filter, which is called the weighted sum. After the convolutional layer the sums are inputted into the pooling layer which aggregates the values provided. There might be multiple instances of these pairs of convolution and pooling layers after each other. Finally, the fully connected layer performs the classification of the values created from the final convolutional and pooling layer. [10], [9] A traditional

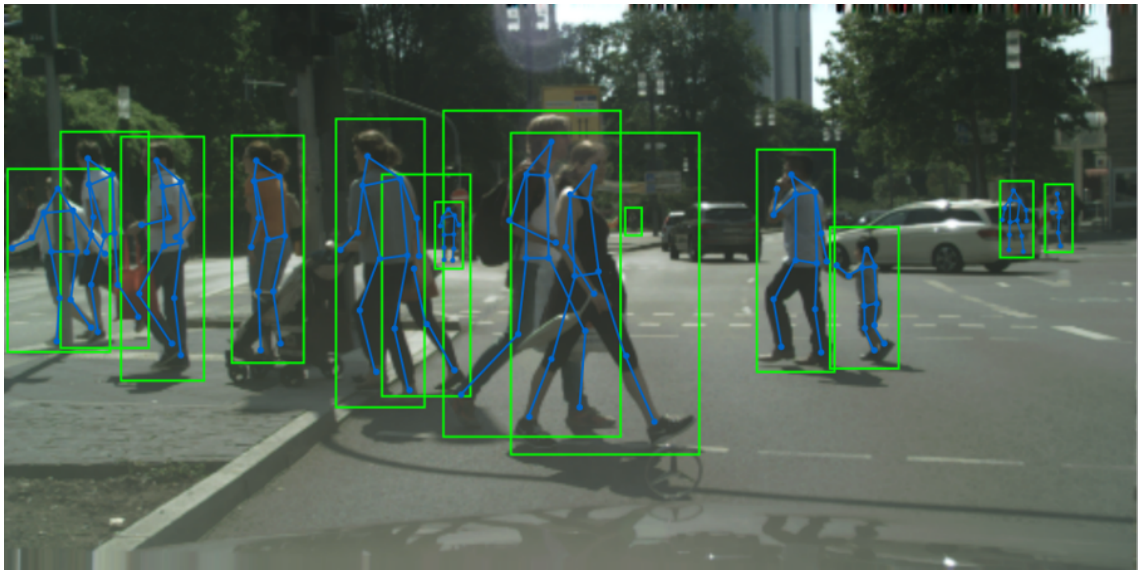

Figure 2.5: Result from a pedestrian detection [8]

CNN is visible in Figure 2.6. In this neural network example, the network includes a flatten layer, which modifies the input of the pooling so that it can be inputted into the fully connected layer.

CNNs are typically used for object detection from images but are not as well suited for natural language processing (NLP) tasks, such as text recognition [9]. Recurrent neural networks (RNNs) are another type of neural network, which is better suited for NLP tasks. CNNs function by going through the neural network once, but RNNs bypass this limitation by creating a type of memory in the neural network, which remember previous states. With this memory RNNs are able remember previous information and take that into account when predicting values. Another distinct feature of RNNs is that instead of adding weights to each node, as in CNNs, they share the weights across layers. RNNs has limitations with long inputs where they encounter vanishing and exploding gradients, where the long-term dependencies are hard to remember. [11] RNNs and CNNs can be used in combination as in the CRNN model, which feeds the outputs from convolutional layers into recurrent neural networks. [12]

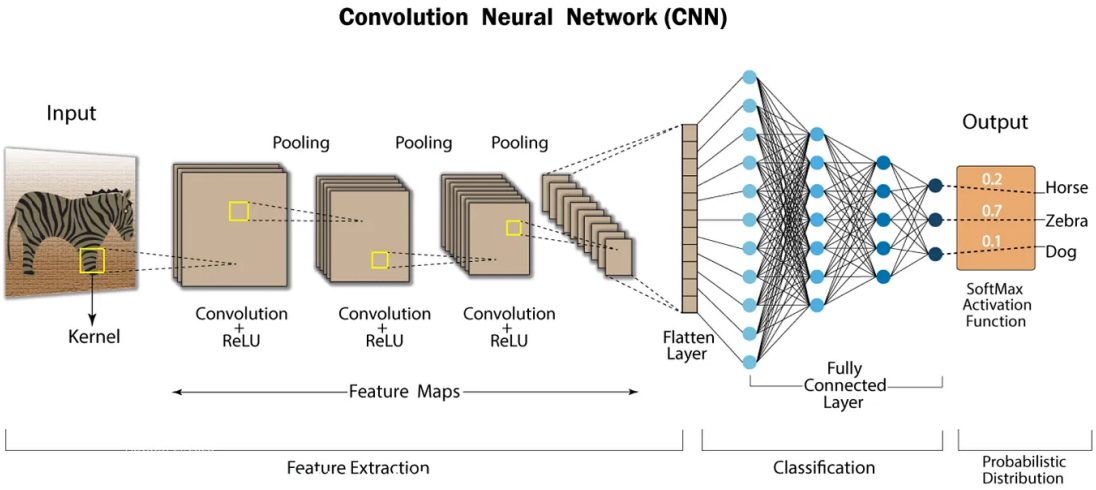

Figure 2.6: Convolutional Neural Network [15]

The latest advancement from RNNs are the transformer models [13]. Transformers are more complex than CNNs or RNNs and they rely on the self-attention mechanism with which the neural network can detect relations between the input. Self-attention works by converting input data into vector embeddings that represent the meaning of the input. The model can then calculate the similarities, correlation, and other dependencies between vectors. With these values the model can emphasize or de-emphasize the importance of input elements at different times. The attention mechanism enables the model to remember long and short dependencies and parallel processing, because they process whole elements concurrently. Compared to the sequential processing of CNNs and RNNs it improves processing speeds and longer inputs are easier to process. This makes transformers superior to previous implementations and they are used prominently in NLP tasks. [14], [13]

Most neural networks are pretrained, meaning that the weights that they inhabit have been trained on large general datasets. Pretrained neural networks are great for general use tasks or instances in which training data is not available. Pretrained neural networks can also be finetuned. In finetuning a smaller dataset is used for modifying a pretrained model for a specific task. Training models require vast

amounts of data, which makes finetuning a great option as it requires far less data. [16] Large language models (LLMs) is a term used for large pretrained models that utilize the transformers model [17] .

Compared to traditional neural networks computer vision problems, especially text detection and recognition, encounter a unique problem of data being unsegmented, meaning there is no direct alignment with the data and results. To solve problems such as these, training processes often implement different algorithms to enhance the training. Connectionist Temporal Classification (CTC) is a common algorithm used in the training of RNN models to solve the problem of unsegmented data. CTC allows the model to align the data and results during the training. [18] Engineering drawings are also lacking in the availability of datasets that could be utilized for training. This makes fine-tuning of models for specially engineering drawing detection hard [19].

2.3 Optical character recognition

Optical character recognition (OCR) is the process of converting textual data from images into machine-readable formats. OCR is prominently used in computer vision but has usages across the field of computer science. Use cases for OCR span from detecting license plates from images of vehicles to machine translation [20]. The OCR process is divided into two sections of text detection and text extraction, where text detection is tasked with locating the textual data and text extraction will transform the text into concrete textual content. Besides these two sections OCR often includes pre- and post-processing in which the input and output data is modified to enhance the results. Traditional OCR models expect the textual data to be easy to find and see. In complex situations, as in engineering drawings, traditional methods struggle to work efficiently. The latest text recognition models are based on neural networks, which are better suited than the previous traditional models.

[20], [21] Figure 2.7 details the OCR pipeline with possible technologies used in each step. The following sections will provide details on each possible technology.

In addition to complex scenes, the engineering drawings in this case inhabit also other variables that might hinder the OCR process. The main issues are that the textual information in the documents might vary heavily. As stated in Section 2.1 the data might be in varying fonts, written by hand, or have errors caused by the scanning. The textual data itself is also problematic as the codes do not resemble natural language. Codes can be a random sum of letters and characters and thus the output of the OCR model cannot be predicted by natural language rules. All these variables hinder the performance of the OCR.

2.3.1 Text detection

Text detection is the process in which textual data is located from an image. During this step the text is located and extracted into its own image, which contains only the detected text. This smaller image can then be used further in other processes. The results from a text detection model are visible in Figure 2.8, where the detected areas are marked visible with green colouring. This detection result can be considered nearly optimal as it has detected only textual content and the whole words. However, this is not always the case as the model might confuse non textual content as text, combine multiple words into the same word, and as is the case in this nearly optimal detection it did miss some textual content from the image. In cases such as these the results can be optimized in post-processing. Text detection technologies are often categorized into classical machine learning-based and deep learning-based. Classical methods are being pushed out by the deep learning-based methods as they are faster and more flexible. Deep learning-based methods can be split further into regression-based and segmentation-based models. The difference between these technologies is that regression-based models handle detection directly from text and segmentation-

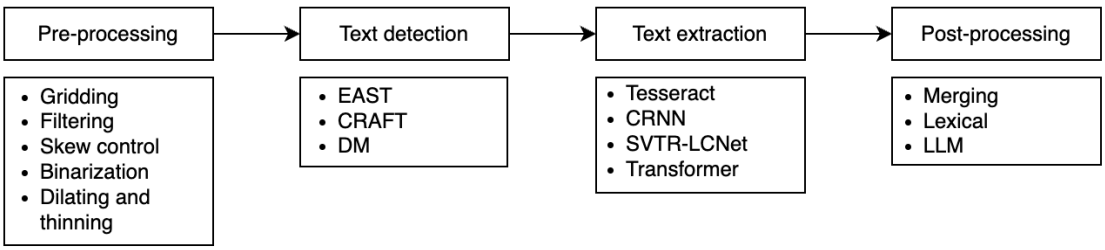

Figure 2.7: The whole OCR pipeline with possible techonologies

based models try to detect text at the pixel level. [22] Two popular models used in text detection for engineering documents are EAST and CRAFT, where EAST is regression-based, and CRAFT is segmentation-based.

Efficient and Accurate Scene Text Detector (EAST) is a model proposed by Zhou et al [23]. The EAST model utilizes a fully convolutional network (FCN) and a Non-Maximum Suppression (NMS). The FCN produces word and text-line predictions, which are then inputted into the NMS to produce results. This concise and simple pipeline results in fast and efficient text detection. However, EAST has limitations in detecting long and vertical texts.

Character Region Awareness For Text Detection (CRAFT) is another model that was proposed by Baek et al [24]. Similar to EAST, CRAFT also utilizes an FCN, but compared to EAST which provided word and text-line predictions, CRAFT produces character region and affinity scores. With the region score individual characters are localized and with affinity score these characters are grouped together. CRAFT can detect curved and horizontal text as the detection is not limited by boxes. However, CRAFT struggles with characters that are close to each other. [20]

Differentiable Binarization (DB) [25] is another segmentation model that is being used. DB is a segmentation-based detection model such as CRAFT and has proven great results and is especially light-weight, which is beneficial for the use-case in this thesis. Compared to CRAFT it does not use complex affinity scores to calculate groups but instead uses the segmentation network for simpler grouping. It has lim-

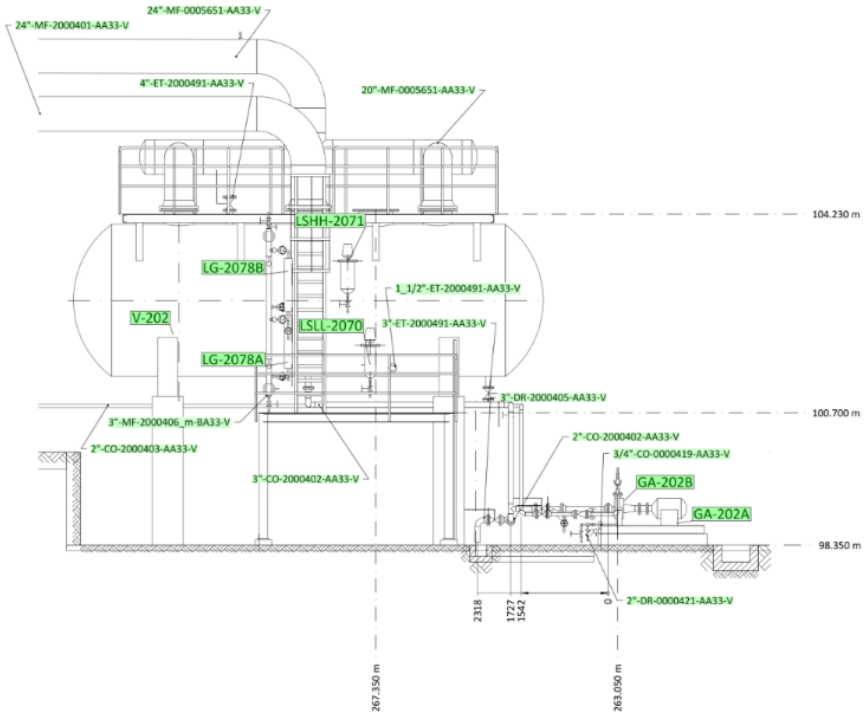

Figure 2.8: Results from text detection visualized

itations in detecting textual information in text-in-text cases, where text is located inside other instances. [25]

2.3.2 Text extraction

Text extraction, also called text recognition, is the second part of the OCR pipeline. After text detection, the extracted images are inputted into text extraction models, which in turn will extract the textual data from those images. Once again classic text extraction technologies have been pushed out by superior deep learning-based solutions, which this thesis will also focus on. Text extraction models can be split based on the algorithm used, from which this thesis will mainly focus on CTC and transformer-based models. [20]

Text extraction models that utilize Connectionist Temporal Classification (CTC) [26], which is a scoring function, have been in use for years, but they are still

prominently used in different OCR solutions. A popular model that utilizes CTC is Convolutional Recurrent Neural Network (CRNN). CRNN was proposed by Shi et al. [12] and combines a CNN, RNN, and CTC to solve text extraction. The model includes Long-Short Term Memory (LSTM) [27] as the RNN, which solves range of context issues of the traditional RNN. LSTM includes memory cells into RNN that enables longer context and makes long-range dependencies possible. This model proposed by Shi et al. is being used as a base in popular OCR solutions such as Keras-OCR [28], EasyOCR [29] and PaddleOCR [30]. Another popular solution is Tesseract [31], which utilizes CTC with only LSTM [32].

Transformers have been developing rapidly and proving great results in text recognition tasks. They excel at long-dependency modelling in which CNNs and RNNs have limitations. [20] TrOCR [33] is a popular OCR model that utilizes transformers instead of relying on a CNN backbone. TrOCR contains an image transformer for visual features and a text transformer for language modelling. It utilizes the encoder-decoder structure where the encoder is an image transformer, and the decoder is the text transformer. The Encoder obtains representations, and the decoder generates the words. [33] There are also models that use a combination of CNNs and transformers such as the SVTR-HGNet [34], which is an implementation that utilizes CNNs as a backbone and a visual transformer for self-attention [34].

2.3.3 Pre- and post-processing

The OCR process contains text detection and text extraction, but the performance of OCR models can be enhanced with pre-processing and post-processing. OCR performance is highly dependent on the quality of data that is inputted to the models. Pre-processing refers to the modifications that are made to the data before it is inserted into the OCR and has the goal of enhancing the quality of the data to

maximize results from the OCR. Post-processing refers to all the modifications done after the OCR model. This means that pre-processing handles the initial documents, and post-processing is conducted with the textual data that the OCR produced. Post-processing is an important step as even with pristine data the OCR might make small mistakes that can be fixed. In addition to pre- and post-processing, there can also be intermediate processing in which there is processing between the results from the detection and before the recognition. This can be utilized to enhance the results from the detection for recognition.

Engineering drawings that have been scanned might suffer from several irregularities. The scanning might have not been perfect and resulted in images that are skewed, blurred, or include noise. These are especially apparent on older scans of engineering documents as the technology for scanning was not as sophisticated as it is nowadays. [35] These imperfections in the documents will affect the OCR negatively. Several different technologies have been proposed for pre-processing the images, which include skew detection and correction, noise filtering, binarization, and morphological filters. [36]

In skew detection and correction, the scanned image is not horizontally aligned, which would make the detection for the OCR more difficult. Skew correction will align the image horizontally to maximize the effectiveness of the OCR. Noise filtering or noise removal is the process of removing unwanted noise from the scanned documents. Noise filtering is done through different algorithms to achieve the desired result. Gaussian filter is a typical algorithm used for filtering noise, where the gaussian distribution is used to remove noise from the documents [37]. Another type of noise removal is smoothing in which values for image pixels are calculated by the sum of the surrounding pixels. Binarization refers to the process of turning a scanned document into black and white. Instead of using grayscale values from 0 to 256 only values 0 or 1 will be used. This removes additional noise from the

images. The calculation for the values are done via thresholding where a value is selected from which pixels above are 1 and below are 0. Morphological filters are filters that transform the textual data in the document. Examples of morphological filters are dilating and thinning. In dilating lines in the image are expanded to be larger, which can help with detecting thin lines, and in thinning the lines are made thinner. [38]

Engineering drawings might also encounter a unique issue of sizing as the size of them is often larger than normal images or what OCR models can handle. This is why gridding, also known as patching, is used. In gridding the image is split into smaller parts and these smaller images are then inputted to the OCR. This process has problems with cutting off part numbers and the OCR detecting them incorrectly. This can partly be fixed by overlapping the images so that the part numbers are not cut off, but this might cause duplicate detections. [39]

An example pre-process is visible in Figure 2.9. This example utilizes the previous explained four processess of translation, binarization, thinning, and noise removal. The first step removes color and translates the image to greyscale, so that the future steps function better. After translation, there still exist shades of grey, which are filtered out with binarization this is visible the second and third step. Next is the thinning, which makes the textual content uniform. The final step is noise removal, which filters out all the obviously unnecessary information from the image. To the human eye the first input image might be clear, but to a machine all the unnecessary information will confuse it. The final result contains only the required information so that the OCR can function properly.

Pre-processing improves the accuracy of OCR models, but they are not optimal. Post-processing methods attempt to improve the results from OCR models after the prediction has been made. There is a vast amount of different post-processing methods, from which typical are merging, lexical approaches, and large language

Figure 2.9: An example of pre-processing. (a) input image; (b) translation to grayscale; (c) binarization; (d) thinning; (e) noise removal [41]

models. Merging is the process of removing repetitions from the results. Results might include duplicate results from detection caused by segmentation, multiple models or other causes. Merging can be done based on the textual results, by location of the detection, or by both. In locational merging the detections are merged if they are nearby each other. Merging is done on a selected threshold, which marks the point in which close-by detections should be merged. [40]

Lexical approaches utilize lexicons and distance vectors to correct mistakes made by OCR models. In these approaches the detected results are compared against a known dictionary with different comparing functions. A traditional comparision function is the Damerau-Levenshtein edit distance, which will compare the minimum number of characters edit operations for an exact match [42]. OCR models also make a lot of invalid detection in which the detected text is obviously invalid or non-existent. These can be filtered without a known dictionary by, for example, filtering

out results that are empty or contain only single characters. This is an important lexical approach to avoid bloated results. [40] LLMs are used for detecting errors and to suggest corrections for mistakes in scanned documents. Especially fine-tuned models have proven to be useful for correcting errors with detections. [43] However, LLMs are heavy and require a lot of additional processing if implemented to the pipeline.

3 Case: eShare

3.1 Background

Cadmatic is a software company that provides CAD and engineering and information management applications. Solutions by Cadmatic are being used in a wide range of engineering tasks and the main users come from marine, process and industry, and construction. eShare is an information management solution developed by Cadmatic. eShare is a digital twin software that is utilized to manage and present information from design projects with main areas of use being in shipyards, small- to large-scale design companies, and oil, gas or chemical plants. The userbase of eShare consists of engineers, but also of management personnel.

eShare is an information management solution, which means that it is not used for the design of 3D models, but as a complementary solution for them. It handles all the information that is connected to the design projects. Design projects include valuable information dispersed on different formats and platforms. Information is located on the model itself, but also on documents connected to the model, and even on database systems. Documents that are based on the model contain engineering drawings, which are the main focus for this thesis, but also other documents such as manuals, bill of materials (BOMs), and reports. Each of these contain information that can be linked to and from the model. Even on small scale systems the amount of data can become cumbersome to control and navigate through. eShare is a so-

lution that solves this issue by reading information from all these external sources and providing a central platform for controlling all the data for the project. This information is then shown on eShare as a centralized platform where the users can easily interact with the model and other information.

A key part for this thesis is the document-model linkage in eShare. Design projects often contain parts and systems that have engineering documents which are based to them. These documents are often independent from the actual 3D model and in some cases even by completely independent actors from the base model. When combining large amounts of documents from various sources and a 3D model that is connected to them, the linkage of these documents to the model can be tedious and difficult to achieve if done by hand. eShare solves this issue by automatically creating bidirectional links between the documents and the relevant parts in the 3D model. When configured correctly, users can traverse between documents and models easily by links that are embedded into the software. For this automatic document-model linkage to work, eShare requires documents that have metadata information of the document attached to them. This metadata information must contain the textual content and location of the text for the functionality to work as intended. In Figure 3.1 is an example of a model-document link, where on the left is a snippet of an engineering drawing that contains links marked with blue. When clicking the circled link, the user will be redirected to the right view in which the model of the engineering drawings is in the 3D model. Vice versa on the bottom right is the documents in which the model is. When clicking those the user will be directed to the engineering document.

3.2 Current solution and limitations

The current solution for document to model linkage in eShare is based on the metadata of files being used. Commonly PDF files contain the textual content in the

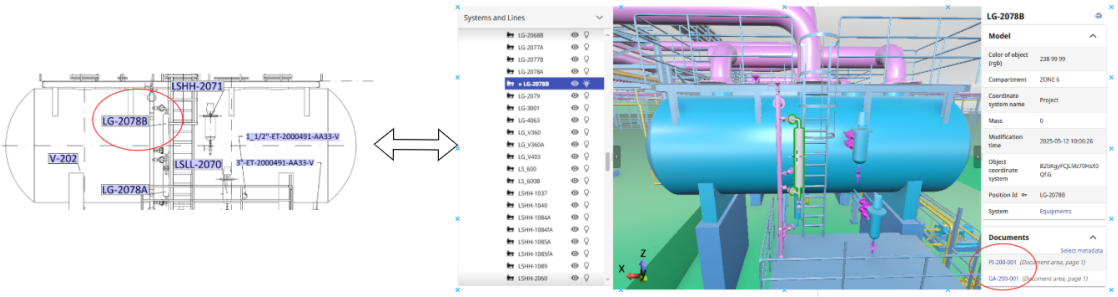

Figure 3.1: A document link between an engineering drawing and the model

metadata of the file, which can be used for different methods, such as searching the document. This metadata commonly contains the text strings and the locations of those strings. The text content is often stored on word-level, meaning that each word is its own entity which are then grouped together when extracting the textual content from the PDF. This makes the accuracy of the metadata dependent on the library utilized for extracting the textual content. eShare contains a custom solution, which will group nearby strings together and performs the best when text from PDFs is extracted on the word-level. In addition to document-model linking, where the data required by the linking is just achieved by querying the metadata of the file itself, this data is utilized for searching the document.

Currently the workflow of injecting documents is as follows. The user first enables document-model linkage by configuring it in eShare's settings. Here the user also schedules if the linkage is executed automatically during a specific time. For example, the user can enable link injection to be executed every Sunday at 12pm, which means that once a week the links in all documents are updated. However, in addition to this, link injection is also executed when documents are opened. When users view a document, it will execute the same process that it would in the scheduled update, which will result in only up-to-date information being visible for users. When the documents are linked, the found links are saved to a database, from which they are queried to show links between the model and documents. Only the found

links are saved to the database, and the textual content is not stored anywhere.

The document-model linkage itself varies between user configuration as the user can configure what the document-model linkage considers as a link and what not. A typical configuration, that the user has set up, is that it will match against known identification values from model objects, smart points, that the user has set to mark points of interest in the model, and regular expressions. In a case such as that, the system will first query the document into metadata information, that will contain all the text lines from the document. These text lines are then compared against the set options and in case of a match, a link from the document is made to the matched object. Here the custom grouping is utilized to match found word-level text strings effectively against multi- and single-word identifications. The created link will then direct the user to the found object on click.

As has been stated in this thesis, the main limitation of this solution is that it can only utilize textual content that is referred in the metadata for the files. In cases where the metadata is non-existent, such as the scenario with older documents, the document-model linkage nor the in-document search will not work at all. Another limitation of the current solution is that the software that reads metadata from the PDF is not perfect. It can accurately get correct textual content but might make mistakes in combining or separating text strings from each other. An example of a common mistake that the OCR makes is visible in Figure 3.2. There the grouping algorithm cannot group the quotation marks and dashes to the word and causes an incorrect word to be created.

In addition to this, the current implementation brings challenges in the implementation of an OCR. The first issue is that the document is injected upon open, which means that if an OCR was implemented into the existing system, it would need to be quick to not cause extended wait times. Another issue is that only found links are saved to the database. This becomes problematic as to get information from

Figure 3.2: Example issue of incorrect detection

the metadata or OCR the pipelines must be ran fully. This once again causes computational overhead and if this implementation is to be kept, requires a lightweight OCR that does not cause processing overhead for running often. Finally, eShare is an on-premises solution for companies, which means that it relies on their systems and in cases might be low on storage, computational power, or both. This requires the OCR to be simple and small. From this the following list of prioritisations can be found:

1. Required computational resources
2. Accuracy
3. Speed

It will be important for the OCR to find a balance between these three variables to provide the best usability and fit for the system.

4 Previous implementations

The following chapter contains information about the research done for this thesis and the results that the research provided. It begings with Section that describes the objectives of the literature review. After the objectives, the methodology itself is explained and the metrics on which the previous implementations were selected. The Chapter concludes by introducing each paper and the solution that they utilized for their implementations.

4.1 Research objectives

There have been multiple different approaches for detecting information from engineering drawings. In this section, four different approaches that have been conceptualized for detecting part numbers and textual information from engineering drawings are presented. It answers the first part of the RQ1 by introducing solutions that have been proposed by previous authors. The main interest of this literature review is to get information and metrics on the detection and recognition models that have been used in similar use-cases to eShare. In addition to this, pre- and post-processing steps that have been utilized are of interest. The previous solutions are introduced in this chapter and in Chapter 5 the solutions are evaluated on a dataset that contains engineering documents that are typically found in eShare. The solutions are evaluated on different metrics, which will give the result to the second part of the RQ1. With this information, the optimal solution for eShare can

be constructed.

4.2 Research methodology

This part of the thesis was done as a literature review. To gather data for the research, popular article databases were utilized. These databases are ACM, IEEE, Springer Link, and Google Scholar. As Google Scholar is a search engine for multiple sources, the search results might contain duplicates and articles from other publishers. The following search query was constructed to perform the search on the databases.

> ("number" OR "ID" OR "identification" OR "label") AND ("OCR" OR "optical character recognition") AND ("engineering drawing" OR "engineering document" OR "technical drawing" OR "technical document")

The search query prioritized especially articles that focused on detecting part numbers from the engineering drawings as it is the primary goal of this thesis. This was done because a lot of research has been done to OCR in engineering drawing but focusing on for example only text referencing the measurements or tables. This query proved to be efficient to filter out these results but contains results that focus on OCR of the whole document. In addition to this search query, the search was limited to only articles in English and articles published after 2021. Such strict timescale was selected because the technology behind OCR is developing rapidly, and information becomes obsolete quickly. The initial search query resulted in 467 articles. These articles were taken for the second iteration, where articles that are obviously not valid for this thesis are filtered. This conclusion was made by reading the title and abstract of the articles. The articles were filtered out if they did not meet the following requirements

- Goal is detecting textual content.

- Focus is on engineering drawings.
- Contains the whole pipeline with detection, recognition, pre- and post-processing technologies realized.
- Utilizes publicly available models, technologies, and data.

With this filtering the results were filtered down to 34 articles, which are all visible in Appendix A.1. Finally, to filter out the remaining articles, the articles were read completely. The articles that were chosen deemed to be the most closely associated with the thesis, contained the most modern and promising solutions, and were peer-reviewed or otherwise deemed trustworthy. After concluding the literature review the Table 4.1 was concluded and contains the four selected pipelines from the previous implementations and answers the first part of RQ1. Chapter 5 utilizes these pipelines to answer the second part of the RQ1 and concludes on which of the solutions is best for eShare.

4.3 P1: Text code recognition and positioning system for engineering drawings of nuclear power equipment

A paper by Ren et al [44] is based on the detection of part identification from nuclear power equipment engineering drawings. The process in this paper is split into two where the first part handles DWG files and the second part PDF files with OCR. Only the second part is of interest for this thesis. The process utilizes the EAST model for text detection and CRNN for text extraction, specially the PaddleOCR library is utilized. In this paper pre- and post-processing is used to boost the performance of the OCR model. For pre-processing before the text detection

gridding is done to split the documents into smaller images. No more pre-processing is done for text detection, but for text extraction noise filtering is used. The image is also embedded into a blank background to detect text on edges. As post-processing they utilized lexical processes. They utilized precise and fuzzy regular expressions in which precise requires the exact match and fuzzy matching is more flexible. They did not provide results for the whole pipeline, but the detection reached 90% accuracy and the extraction varied more heavily from 60 to 90% when compared to ground truths.

4.4 P2: Text detection and post-OCR correction in engineering documents

Francois et al. [45] provide the next pipeline in their paper. Their paper focuses on detection of all textual data that also includes part numbers. For the text detection they utilized EAST but compared to other implementations they fine-tuned it with engineering documents and removed the NMS layer to prevent it from cutting long strings. For text recognition they used the popular Tesseract OCR that was configured to extract single-line values. For pre-processing they utilized binarization using Otzu thresholding and creating duplicate 90-degree orientated versions of the original documents to detect vertical text. In addition to these they also utilized gridding with a 4x4 grid. For post-processing they applied merging to the results of the EAST model by combining close pairs and splitting in case of multiple values close by. This resulted in false positives, which were handled by filtering empty characters. Post-processing from the OCR was done with a unique affinity propagation clustering, where the tags are merged by clustering them with edit distances. They were able to reach 82% ground truth detection rate.

4.5 P3: Leveraging transformer-based OCR model with generative data augmentation for engineering document recognition

Khallouli et al. [46] valuate different OCR solutions for extracting text from legacy engineering drawings. Instead of using the EAST for text detection they decided to utilize CRAFT. For text extraction they utilized CRNN and transformer-based solutions. They included pretrained and fine-tuned models that were fine-tuned on their augmented data that they created themselves. They utilized Tesseract, CRNN-based Keras-OCR and EasyOCR, and transformer-based TrOCR. They used fine-tuning as their dataset included varying fonts, which proved hard to detect for pre-trained models. For pre-processing they implemented traditional technologies which included binarization, noise filtering, and scaling in which the documents are unified in size. Interestingly, the paper did not utilize any post-processing in their approach. They concluded that fine-tuning improved the performance of the OCR significally and the fine-tuned transformer model reaching the highest results at 3.94% error rate on words against the ground truth. The base transformer model reached 12.47%.

4.6 P4: Artificial intelligence-based solution for the automatic extraction of pipeline metadata from piping and instrumentation diagrams (P&IDs)

A new paper that has not yet been peer-reviewed from Shteriyanov et al. [47] provided a pipeline that produced impressive results against EAST and CRAFT-based pipelines in detecting piping information from P&ID. For text detection they

Table 4.1: Grouping of previous implementations for part identification from engineering drawings

Author	Text detection	Text extraction	Pre-processing	Post-processing
Ren et al. [44]	EAST	CRNN (PaddleOCR)	Gridding Noise filtering Padding	Lexical
Francois et al. [45]	EAST	Tesseract	Gridding Orientation duplicates	Merging (Clustering)
Khallouli et al. [46]	CRAFT	Transformers (TrOCR)	Binarization Noise filtering	None
Shteriyanov et al. [47]	PSENet	SVTR-LCNet (PaddleOCR)	Gridding	Merging Lexical

proposed a Progressive Scale Expansion Network (PSENet). The PSENet is a segmentation-based model such as CRAFT and DB, but is exceptional in detecting closely grouped text and oriented text [48]. For text recognition they utilized SVTR-LCNet in PP-OCRv4. For preprocessing they utilized gridding with overlap that is then merged in the post-processing. They also included post-processing correction into the OCR in which typical errors and domain specific correction rules were applied. Their solution proved a 95.14% recall. [47]

5 Evaluation and optimization

5.1 Evaluation scenario

Compared to previous implementations there is a flurry of differences as the goal is to determine if the solution is suitable for a business implementation such as eShare. This thesis will utilize different evaluations, datasets, and overall methodology to determine the best pipeline. The main goal of this test is to find the best solution for eShare's use-case and to evaluate on the possibility of real-time detection for large engineering drawings. The main differences from eShare to previous solutions is that eShare requires a solution that works with computationally limited power, a varied dataset that contains data from multiple fields of engineering, and a solution that will recognize on the word-level, not sentence-level. Meaning that detections are individual words and not sentences.

eShare is often deployed on a platform that does not have a lot of computational power and often does not have GPUs available that could be utilized for OCR. This forces the thesis to investigate solutions that will only utilize CPUs and are computationally less heavy than the state-of-the-art solution. In addition to this, previous implementations are focused on a singular field of engineering in which the dataset is dominated by that field. In eShare the documents can be from a variety of fields and sources, which needs to be considered in the testing. Documents from differing sources causes the documents to be in different forms in terms of layout,

structure, and utilized symbols. In addition to this the font used varies heavily between sources. Previous solutions also attempted to detect text on a sentence-level, meaning that detections could have spaces in them. In the case of eShare, only word-level detections are needed as there already exists a solution to merge word-level detection to part numbers. Finally, previous solutions focused only on accuracy in the results and trying to provide the state-of-the art model. However, in eShare the duration and F1 accuracy, which will be explained in Section 5.1.4, are important. Thus, the evaluation will be different and focus on different parameters.

5.1.1 Models

The systems that will be utilized in testing are visible in Table 5.1. The systems S1-S3 are from previous implentations, but the CRAFT has been changed to EAST with TrOCR. This is done because CRAFT is known to be computationally heavy and is known already beforehand that it would not suit the use-case for eShare. EAST on the other hand is lighter and has more potential in achieving the required speed and accuracy. Ren et al. [44] and Shteriyanov et al. [47] stated in their papers that they also utilized PaddleOCR. Ren et al. utilized it for EAST and CRRN and Shteriyanov et al. for the PP-OCRv4. Since the publication of these papers PaddleOCR has released a new pipeline, PP-OCRv5, which will be used instead as it proves to beat the previous pipelines on accuracy, especially on the smaller models [49]. PaddleOCR was chosen as the main provider as it was utilized in the previous implementations, has been proven to give great results, and provides a wide variety of models that are different in size and pre-processing options. [50], [51], [52] For this paper to adequately find the best model a combination between the previous implementations are also provided. These systems contain the DB & Tesseract, PP-HGNetV2 & Tesseract, and DB & SVTR-HGNet combinations. DB & SVTR-HGNet is implemented via the older PP-OCR pipeline that utilized the DB

Table 5.1: Models used for experiments

System number	Text detection	Text extraction Read model	pipeline
S1	EAST	Tesseract	Own
S2	EAST	TrOCR	Own
S3	DB	TrOCR	Own
S4	DB	Tesseract	Own
S5	DB	SVTR-HGNet	PP-OCRv5 (with older PP-OCR)
S6	PP-HGNetV2	SVTR-HGNet	PP-OCRv5
S7	PP-HGNetV2	Tesseract	Own

for detection, but fed into the PP-OCRv5 for recognition. With these combinations each detection and recognition model is inspected and the optimal choice for eShare will be found. Once again it is important to note that all the models are CPU-based and will use the smallest provided models to maximize the fit for eShare.

5.1.2 Pre- and post-processing

To achieve results that indicate the performance of the models and no other processes, similar pre- and post-processing is used with all the systems. However, as the models support different functionality, two different pipelines are used to maximize output from the models by utilizing model-specific methods. The used pipelines are visible in Table 5.1. All pipelines contain the same functionality in which the pipeline will start with a PDF file and output text lines and coordinates for those text lines. Each pipeline also contains the same greyscale transformation, filters, and gridding sizing. The largest changes are in the intermediate processing that is between text detection and text extraction and functionality on how this is configured.

The pipeline labelled *Own* is the most extensive and contains most self-made code. The whole process is visible in Figure 5.1. First the document's optimal DPI and patch count is calculated, and morphological filters are applied. The image is translated into grayscale and binarized to allow only pure black and white values.

The previous solutions did not provide reasoning on what resolution they performed detection and recognition, but DPI between 200 and 400 deemed to be the optimal. For patch size, sizing from 2x2 to 4x4 is chosen. Both values are chosen based on the size of the document, where for example A4 sized documents are split into 2x2 with 200 DPI and A0 or larger documents are split into 4x4 with 400 DPI. This ensures that even small text is visible on larger A0 sized documents, but additional processing is not required on smaller A4 sized documents where small text is often not apparent. Each patch contains 5% overlap with the other patches to minimize the effect of patching.

After the patches are detected, each detected box from the edge of the patches is merged into each other so that if there was a box on side-by-side patch edges, it will be merged into a single detection. After this, all boxes are deskewed so that the textual content is horizontally aligned. Finally, the individual detections are inputted for text recognition, from where results are observed. It is important to note that there are conversions between bytes and images between the processess as some take in bytes and others image files.

The second pipeline labelled *PP-OCRv5* is the latest pipeline from PaddleOCR. The pipeline contains a lot of individual processes, which are disabled for this thesis to align best with the other pipeline. Thus, the main difference between the pipelines is that the PaddleOCR will utilize its own deskewing functionality and not the custom solution. [49] The pre- and post-processing functions are the same for both. The post-processing is the same as in the previous solutions. It will only utilize traditional lexical filters for the processing. The previous solutions did not go into depth on what lexical filters they utilized, and a custom solution was created for this thesis. In it, some cleanup is done, such as removal of multiple spaces and a whitelisted list of characters is only allowed to appear. In addition to this, the detections are split based on spaces to ensure word-level detections.

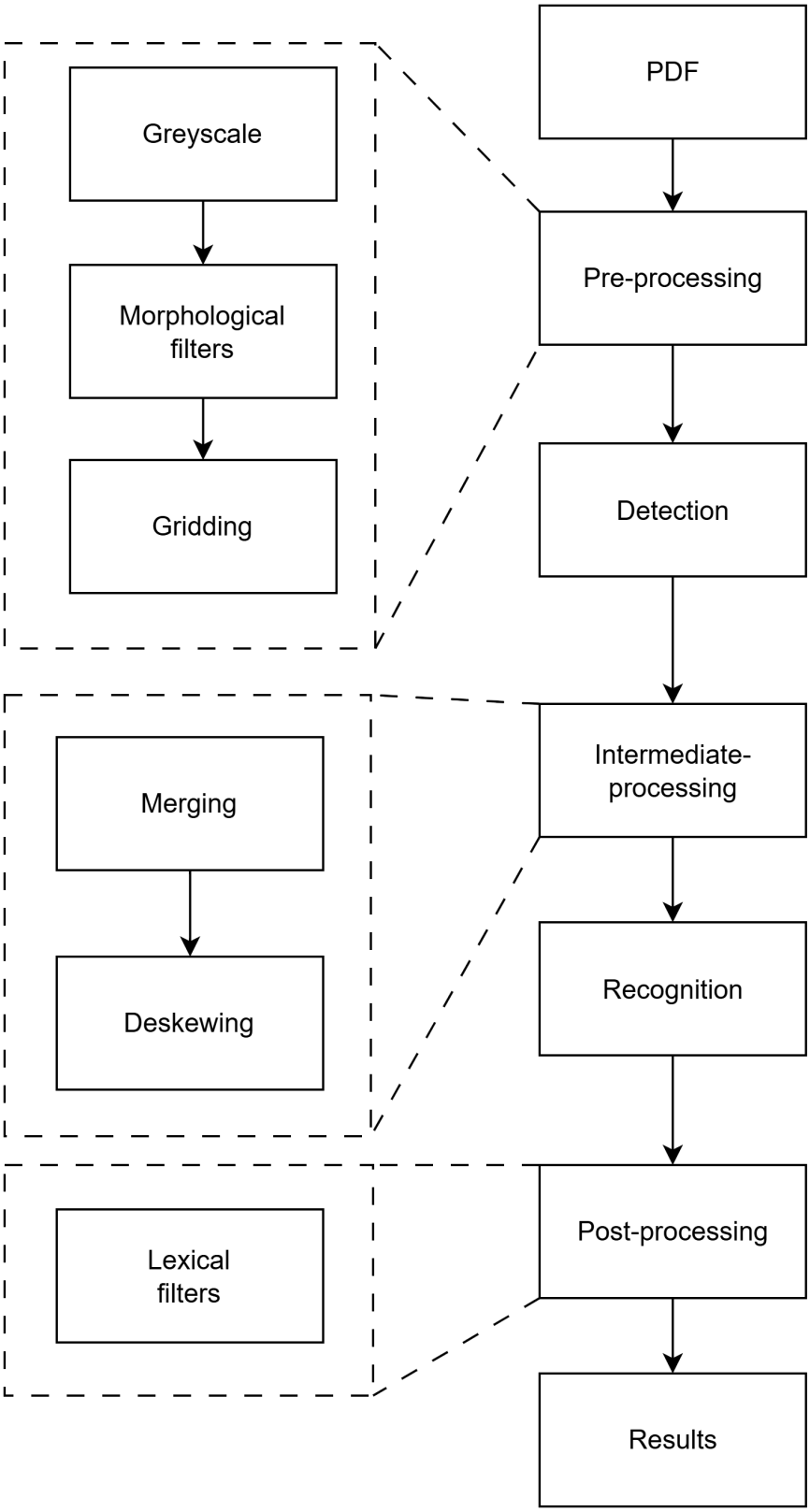

Figure 5.1: Own pipeline for OCR

5.1.3 Dataset

eShare supports a wide range of engineering drawings, which are different to previous implementations as they have been from a single field of engineering. Instead of focusing on a singular type of engineering drawings, the test dataset will contain a varied set of engineering drawings. The dataset is acquired from test data used at Cadmatic and drawings acquired from customers. As the data in the documents is classified, it cannot be used for finetuning or publicized. This also is the major reason as to why only pretrained models are used in the thesis. All the documents in the dataset are PDFs that might or might not include metadata information about the textual content.

The dataset is split into four different sets by the size of the documents. The first set containing smaller documents, which are A3 or smaller in size. The second set contains documents between A2 and A1 and the final set contains documents which are A0 or larger. The split between the sizes is visible in Table 5.2 (a). This is done to measure the duration of the detection in different scenarios. The final set contains all the documents in one, which will be used to measure the accuracy of the systems.

The documents which make up the sets are visible in Table 5.2 (b). Each document is unique in one or more ways to the others. This means that it might have differing font, layout, or icons. The dataset contains engineering drawings, but also a few title pages, which are often found in conjunction with engineering drawings. Assembly drawings are most heavily represented in the set as they often contain part number identification, which is the primary focus of this thesis. In total the dataset contains 16414 text strings that are possible to be found from the documents.

Document type	count
P&ID	9
Assembly	21
Isometric	6
General plan	3
Listing	4
Title page	6
Total	49

(a) Split between document type.

Document size	count
Small	21
Medium	15
Large	13
Total	49

(b) Split between document size.

Table 5.2: Dataset distribution

5.1.4 Evaluation and running

The evaluation is done on two main parameters of accuracy and speed. The smaller the model the better, as it then requires less storage, consumes less computational power, and is typically faster than heavier models. Duration is the total time it takes from the image to be translated into text strings, which contains the most time-consuming parts, detection and recognition.

Accuracy can be measured by various metrics from which the precision, recall, and F1 score will be used. *Recall* is the percentage of text strings that were correctly detected from the absolute truth. Additional incorrect detections will not affect the precision of the system, but with precision they are taken into account. *Precision* represents how well the system was able to avoid incorrect detections, it is the percentage of correct detections to incorrect detections. Finally, *F1* score is a metric that combines the precision and recall. [53] The mathematical functions to calculate precision, recall, and F1 are visible below. There true positive (TP) is detections that are correctly detected, false positive (FP) is detections that are incorrectly detected to be correct, and false negative (FN) is detections that are detected falsely as negative meaning they are not detected. F1 was deemed to be the best accuracy metric for this thesis as eShare requires accurate detections, but incorrect detections also affect the system negatively. Incorrect detections can clutter the system with unnecessary data and cause incorrect document-model links, causing problems for

users. F1 score takes this into account better than a pure precision accuracy.

$$recall = \frac{TP}{TP + FN}$$

$$precision = \frac{TP}{TP + FP}$$

$$F1 = 2 * \frac{precision * recall}{precision + recall} = \frac{2TP}{2TP + FP + FN}$$

The code for the tests is written in Python and the pipelines are run locally in a system without a dedicated GPU. The tests are run in Windows 10 Pro 64-Bit Build 19045 with Intel Core i7-9850H CPU at 2.6GHz with 6 cores, 32GB of DDR4 RAM, and 1 TB NVMe SSD. The duration of the OCR depends heavily on the system at use, which emphasizes the percentage change between detection duration and not so the actual timing of the detection. However, to achieve real-time detection, the detection should be possible in under 10 seconds on this current system. This was selected as the threshold for real-time detection, because of the system configuration of the test machine. Tests are run multiple times, and the results are an average of the runs. The tests are ran on the large dataset to calculate the accuracy scores and on each of the size split dataset to achieve the duration of detections.

5.2 Results

The test cases were ran with each system to examine the suitability for eShare. The results from the tests and the answer to the second part of RQ2 are visible in Table 5.3 and visualized in Figure 5.2. Each system performed quite well, but there were clear differences between each model. The systems were overall performing somewhat worse than the results which were given in the previous implementations.

However, the models were performing similarly in relation to each other as they were in the previous implementations. The drop in performance is likely from the difficult to understand engineering documents that were utilized in the dataset.

Detection

The three models that were selected for this benchmark were EAST, DB, and PP-HGNetV2. EAST was clearly the worst performing model as both models, Tesseract and TrOCR, provided the worst results compared to each other. Interestingly, the EAST model was the best at detecting the highest recall compared to precision. This can be explained by the fact that the EAST model was able to only detect the easy and clear text-strings from the documents but missed when the data became more complex. This also being the reason why the EAST model performs noticeably worse than the other two models. EAST was also worse than the others at bounding the text in accurate boxes. The boxes would often be inaccurate, which would critically hinder the performance of the recognition model.

The DB and PP-HGNetV2 were able to perform more effectively and proved better results. When comparing the DB and PP-HGNetV2 with Tesseract and SVTR-HGNet it is clear to see that both models performed quite similarly. This can be explained by the fact that both models were able to detect almost all text strings from the documents. This makes the changes between the models minimal and more dependant on the recognition model that is being used. DB was able to achieve high accuracy with TrOCR and PP-HGNetV2 which proves that both models would be great for use in eShare. Compared to EAST, both models were able to detect text from complex scenarios and bound the textual content better. Even though the lower threshold for detecting text caused the larger discrepancy between precision and accuracy, it enhanced the overall accuracy of the model by a large amount. In addition to the accuracy, both models were able to achieve similar

Table 5.3: Results from the tests

System	Models	Recall	Precision	F1
S1	EAST & Tesseract	0.577	0.584	0.581
S2	EAST & TrOCR	0.646	0.635	0.64
S3	DB & TrOCR	0.833	0.733	0.802
S4	DB & Tesseract	0.789	0.72	0.763
S5	DB & SVTR-HGNet	0.834	0.766	0.798
S6	PP-HGNetV2 & SVTR-HGNet	0.839	0.787	0.812
S7	PP-HGNetv2 & Tesseract	0.785	0.707	0.744

duration, which is visible in Table 5.4.

EAST might have possibly achieved similar accuracy if the DPI and patch count would have been raised, but as the model was already the slowest of the three, it would not have been a viable solution. Detection is an especially important step in the pipeline since if the model detects too much and too easily, it will flood the recognition model with unnecessary processing and will exponentially grow the computational time required. This makes it important to keep the confidence levels high on both models and emphasize accurate detections. DB and PP-HGNetV2 were both able to achieve high accuracy, but also caused incorrect detections, which negatively impacted the F1 score.

Recognition

For recognition the following models were used: Tesseract, TrOCR, and SVTR-HGNetV2. However, the detection models were similar in duration, but the recognition models made the larger change, which is visible in Table 5.4. This is explained by the fact that the recognition model had to process each text string and not just the patches, which resulted in much larger processing. Each model caused a jump between medium and large documents, which is explained by the sheer amount of data that the larger documents contained. However, with this data it is visible to see that the TrOCR model was the slowest of the models overall. This was to be expected as it was the most complex and largest model of the three. TrOCR

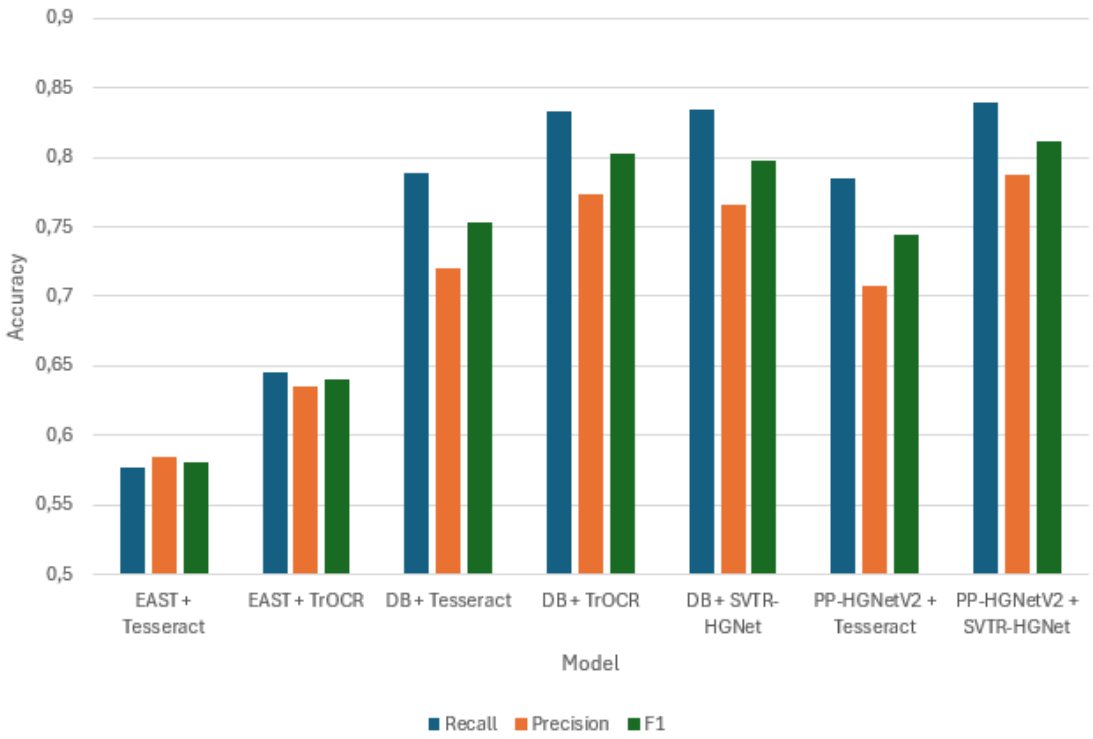

Figure 5.2: Results from the tests visualized

proved great results especially with DB where it was almost able to reach the PP-OCRv5 pipeline. Tesseract and SVTR-HGNet were both more lightweight and able to perform in adequate time compared to the TrOCR and eShare's requirements. Tesseract and SVTR-HGNet were both able to be faster than TrOCR and enabled for their implementation into eShare. However, Tesseract proved to be the less accurate of these two, as SVTR-HGNet noticeably outperformed it on the F1 accuracy. In addition to this the increased accuracy did not add a significant amount of time to the processing, which makes SVTR-HGNet the clear choice for the recognition model. Overall, the reached accuracy of 84% is not optimal and could be enhanced further.

Pipelines

The testing utilized two pipelines, one of which was the PP-OCRv5 pipeline from PaddleOCR and the other was a custom pipeline that was based on the previous implementations. The PP-OCRv5 pipeline was great at handling legacy documents, but at the same time was not easily customized and included unnecessary processing for relatively predictable engineering drawings compared to natural scenes. The pipeline based on the previous solution however lacked in functionality and could be improved further to enhance the performance. Clear pain-points in the pipeline were tables and word-level detections. The detection model would not detect text inside tables, and the recognition model would group individual words together instead of detecting them as separate words. This could be improved with additional pre-processing and intermediate processing. To maximise the results from the detection and recognition models, a custom pipeline must be implemented that combines the strengths of both custom and PP-OCRv5 pipelines.

None of the systems were able to provide real time detection, which can be seen from Table 5.4. It is visible that DB and PP-HGNetV2 in conjunction with Tesseract and SVTR-HGNet, were able to reach near real-time detection in smaller and medium-sized documents, but as soon as the amount of data in the larger documents rose, the systems could not handle it in real time. The amount of data amplified the differences between the recognition models and confirmed that real-time detection would not be possible for eShare. When observing the percentage changes in the average of the systems the DB and SVTR-HGNet (S5) was the best performing system, from which the PP-HGNetV2 and SVTR-HGNet (S6) took 4% slower, meaning that system S6 took 4% more time to process than system S5.

The DB and Tesseract (S4) was also only 14% slower and PP-HGNetV2 and Tesseract (S7) was 28% slower. These were all contained in the similar group as the fastest combinations. The EAST and Tesseract (S1) system being an outlier as

Table 5.4: Average detection duration per document in dataset

System	Models	small	medium	large	All
S1	EAST & Tesseract	12.817	27.649	90.112	39.42
S2	EAST & TrOCR	34.169	54.072	247.704	93.627
S3	DB & TrOCR	17.016	24.521	122.123	48.993
S4	DB & Tesseract	6.812	11.502	52.635	21.687
S5	DB & SVTR-HGNet	7.727	12.984	44.079	18.979
S6	PP-HGNetV2 & SVTR-HGNet	6.811	13.348	46.799	19.746
S7	PP-HGNetv2 & Tesseract	8.905	15.257	58.587	24.393

it was 108% slower than the fastest system, even though it was able to detect the least. Finally, the combinations with TrOCR were 397% slower with EAST (S2) and 159% with DB (S3).

5.3 Result analysis

The results from the tests prove that the current solutions are not as effective for eShare's use-case as they were for their respective solutions. Especially the older EAST model was struggling when handling large and cluttered documents that contained textual content in various formats and in non-ordinary situations. However, the DB and PP-HGNetV2 were both able to function surprisingly well in these cluttered scenarios. Both models did run into some difficulties when detecting highly concentrated individual text strings, but most of the detections in even highly clustered scenarios were correct. Other pain-points were text that was close to edges of circles. Especially with larger documents in which the text is small, these values were missed and only the text from the centre was detected.

Both models, DB and PP-HGNetV2, were quick and effective at handling large amounts of data and were still able to provide accurate results. Both models also functioned surprisingly well on the limited computational power that was present in this test case. However, in addition to the previous hard-to-detect cluster of strings the models had difficulties detecting separate words and combined strings. This was

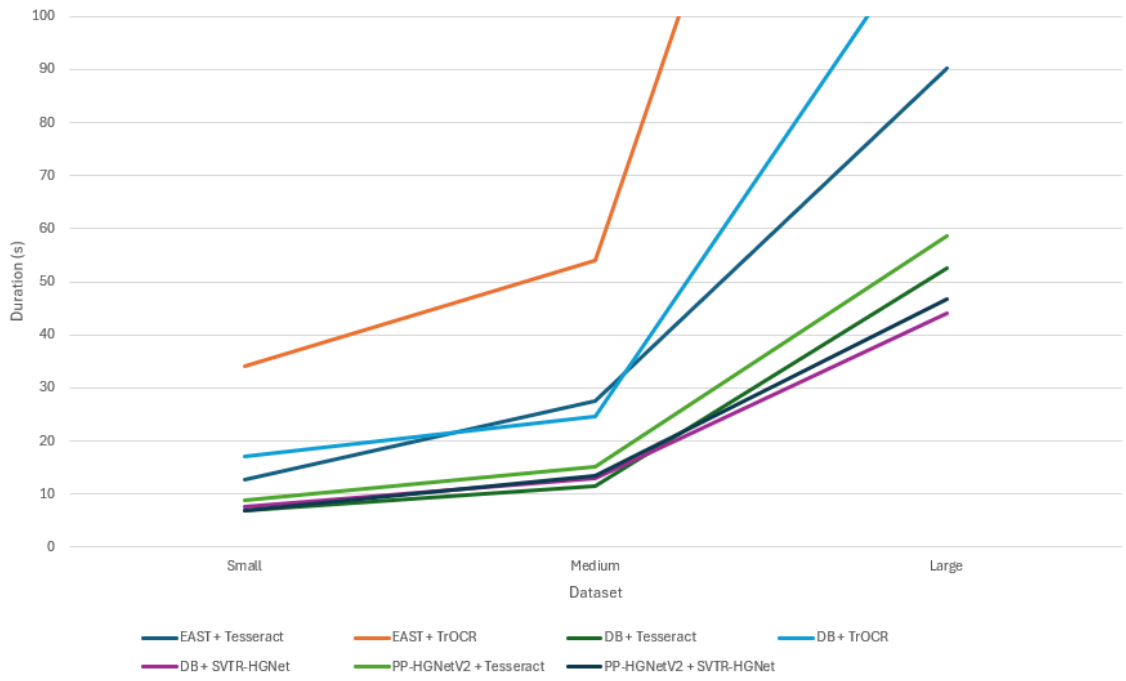

Figure 5.3: Timing of the tests visualized

especially apparent in larger documents and in documents where the font was more confusing. In some cases, there was not a clear split between spaces and whitespaces between letters, such an in documents that were exported such that text was not a font, but rather a combination of shapes. An example of difficult area is visible in Figure 5.4. In the end PP-HGNetV2 was the most accurate of the detection models and proved to be the fastest at providing these accurate results.

For recognition the three models all had their own benefits and clear applications that they could be used for. TrOCR was clearly too complex to be utilized on a CPU only platform and caused significant overhead on the processing. The processing was so slow that it could not be utilized even in a non-real-time application, as running documents in the background too slow would also cause issues and additional overhead to the user if they happened to be using the software at the same time. However, the recall precision ratio that the TrOCR provided was surprisingly good and in a GPU-based solution the larger TrOCR models could pro-

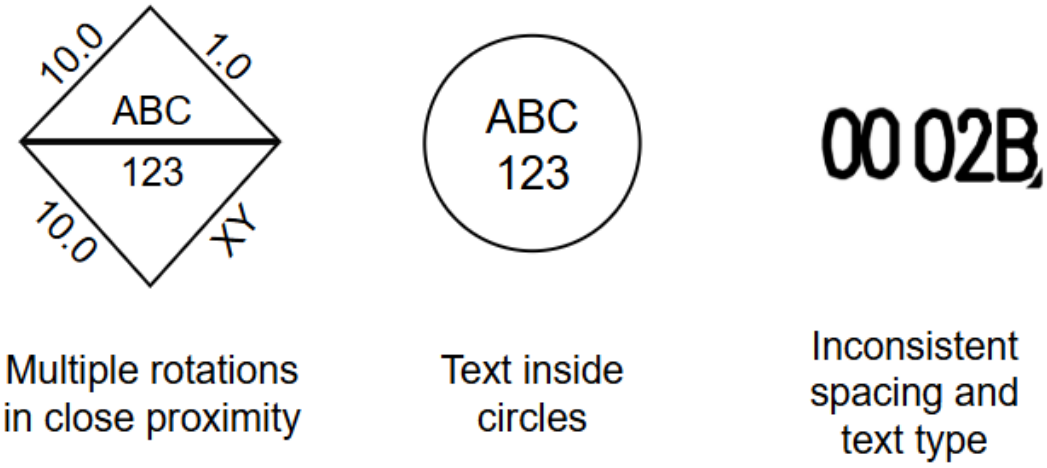

Figure 5.4: Difficult to detect text strings

vide excellent results for the use-case. However, for eShare this is not a possibility. Tesseract on the other hand proved to be a small and lightweight model that was surprisingly effective at recognition. The Tesseract model benefitted especially from the available configurability, which allowed it to be fitted for this task via for example whitelisting only required characters. This allowed the model to detect only the correct characters and mistakes between languages or formulas could not be made. In addition to this, Tesseract could be modified to detect only single lines of texts that improved its performance. Finally, SVTR-HGNet was like Tesseract as it provided great results and was fast enough. It significantly outperformed the Tesseract in the PP-OCRv5 pipeline, but lacked the customization that Tesseract provided and sometimes caused incorrect detections because of this. Even when the PP-OCRv5 pipeline was configured to the English language, the SVTR-HGNet model detected characters from other languages. This caused some issues with the recognitions. However, as the SVTR-HGNet was marginally better than Tesseract, it was the clear winner in the recognition models. In addition to this, PaddleOCR is one of the leading developers in OCR and are developing new models constantly,

which makes it a more attractive choice for the possible future development that could happen and the easy possibility of swapping models in PaddleOCR's library.

Finally, the two pipelines that were tested in this scenario, the custom pipeline based on previous solutions and the PP-OCRv5 pipeline provided by PaddleOCR. Both pipelines were functioning as expected and had their benefits. The PP-OCRv5 pipeline outperformed the custom pipeline but contained a lot of unnecessary processing and lacked the customizability. For the quite simple use-case of engineering documents, the machine learning-based deskewing was unnecessary overhead to the whole process. Engineering documents contain text on a clean background in set rotations. These could easily be detected with lighter algorithmic solutions. The custom pipeline lacked some functionality but allowed full customizability.

The models had a clear optimal choice to be chosen, but with a pipeline a more tailored solution could be better for eShare. The previous solutions did not utilize the customizability and processing that was possible with the formatted structure of engineering documents, and is possibly taken further. A critical point is the word-level detections required by eShare, which caused issues with both pipelines. Both pipelines combined separate words into single strings of data. With the format of engineering drawings, it is known that text is in a specific format and with some image processing the split points for spaces could be calculated. As PP-OCRv5 does not allow custom intermediate processing steps, a custom solution has to be utilized. However, the PP-OCRv5 detection and recognition models could be utilized separately, which would allow the additional step and remove the unnecessary processing that was in PP-OCRv5. Both pipelines contained the same pre- and post-processing that was based on the previous solutions. The pre-processing did sometimes lack, especially with the morphological filters on images. Post-processing was not complex and contained just simple lexical filters to filter through the detections and fix obvious mistakes. Both areas are subject to further development.

Even though DB with Tesseract and PP-OCRv5 were both quite quick on CPU, they could not provide real-time detection. The issue was not so with the detection as it could handle detection of even A0 sized documents in adequate time, but more-so the vast amounts of data in these documents. A single A0 document could include thousands of text strings that the detection models detected. Running each of these strings through the recognition model on CPU was too demanding and created the largest bottleneck. There were no solutions found to circumvent this bottleneck, and thus real-time detection must be excluded from eShare's implementation. Even though real-time detection is not possible, it is crucial to achieve as fast as possible detection, to not cause issues if the detections are ran in the background or if the user wants to open a document with real-time detection.

In conclusion, the PP-OCRv5 pipeline proved to be superior to other solutions provided by previous authors. Especially, the models that PP-OCRv5 uses, PP-HGNetV2 and SVTR-HGNet, are the optimal models for eShare's use-case. However, neither pipeline is optimized for eShare's use-case and the scanning of engineering documents. This means that a customized solution that will improve the accuracy should be implemented to extract the best possible results from the detection and recognition models. The F1 scores on all the models were also quite lacking and require further improvement in filtering out incorrect detections.

5.4 Proposed solution for eShare

eShare and engineering documents provide a unique environment for OCR, especially when compared to natural scenes. The OCR encountered difficult scenarios, which are not encountered in natural scenes or non engineering documents. These were clear weaknesses in the whole pipeline. This could possibly be fixed with finetuning in the case of having available data, but as this is not the case for us, another solution must be proposed. Previous solutions did not utilize large amounts of

pre-processing, intermediate processing, or post-processing techniques that could enhance the performance of the OCR. As engineering documents are predictable in their format, these could be utilized to a greater effect than previously in natural scenes or previous solutions for engineering drawings. In this case, especially the word-level detection proved to be difficult in some scenarios, which could be solved with enhanced processing between the detection and recognition. In this Section an answer to RQ2 is provided by proposing a solution that is a unique and tailored solution for eShare but can possibly enhance the results also in other environments.

Model

The system to be used was chosen based on the previous Section 5.2 that provided valuable information on the performance of differing models. On the metrics that they were compared, PaddleOCR with SVTR-HGNetV2 and PP-HGNetV2 (S6) proved to be the most suitable solution. PaddleOCR also had other benefits that are useful for the implementation, such as that the system is open for modification with possibility for fine tunning in the future, it allows utilization of models by themselves to create custom intermediate processing, and provides a library that eases the implementation of the system. All these reasons make system S6 the best solution for eShare. As none of the systems were fast enough to provide great results for real-time detection with large engineering documents, a caching mechanism needs to be utilized to avoid repetitive processing. Even though real-time detection upon document open is not viable, speed is still a crucial factor as extensive processing times would affect the user negatively if the process is running in the background.

Pipeline

As said in Shapter 5.2 the pipelines in the testing were lacking and could be improved further. The PaddleOCR library will be utilized for detection and recognition,

but the other processing that the PaddleOCR library does is disabled. Instead, customized processing is added to improve the accuracy of the system. The detection and recognition model is accessed separately instead of utilizing the PP-OCRv5 pipeline that was used in the test scenario.

The proposed solution does not differ much from the previous solutions in the pre-processing. It will utilize deskewing to ensure that the document is horizontally aligned and morphological filters were used to make the PDF as clear as possible. For morphological filters, binarization with OTSU thresholding proved to be the best algorithm for creating a binarized image. Legacy engineering documents that were scanned images in PDF and which contained light colours such as yellow became cumbersome as the edges of text were hazy and resulted in incorrect binarization to catch these light colours. To bypass this the PDF was converted into HSL colours which was then binarized. In addition to this, a kernel was used to enhance the edges of text in the images to ensure that the edges were as sharp as possible since zoomed in images caused the pixels on the edge of texts to be hazy. This haziness resulted in incorrect binarization, and the text was too thick. It was also observed that the detection model did not require as high-quality images for processing and thus the patches could be scaled down a bit when detecting text. However, it was important for the patches to be high quality when extracting the detections as the recognition model was more dependant on the quality.

In addition to this, all PDFs are split into patches that overlap with each other and each patch is pre-processed by itself. As eShare requires memory management, loading large engineering documents at appropriate DPI was impossible due to the sizing issues. Thus, each page of PDF is processed so that a single patch is extracted and loaded into memory, processed and then discarded. This way the memory constraints of eShare are bypassed. To handle detection on the edge of patches, the coordinates of them are stored, merged with each other, and then these remaining

patches are rendered straight from the original PDF. After the preprocessing, the patch is fed into the text detection module that utilizes SVTR-HGNetV2. Results from the detection are given for intermediate processing.

The intermediate processing is the part of the pipeline, which contains the most changes and improvements to previous solutions. The processing includes processes found from previous solutions, such as merging closely found boxes and deskewing. However, in addition to these, engineering documents often contain text in closely packed tables and because of how the engineering documents are formatted the contents are easily split into their own detections. Thus, detected images are split vertically and horizontally based on the whitespace found in the image. If the image contains either vertically or horizontally large whitespaces that are abnormal, they are considered as spaces. This solves the issue described in Chapter 5.2 in which varying fonts caused the detection model to mistakenly detect individual words as one. Engineering documents also often contain abnormal word structure that can confuse the model that is constructed on natural language. Due to these reasons, it is beneficial to split the sentences into words on the intermediate processing step and not to trust the text recognition model to correctly detect the spaces. The whole process of this splitting is visible in Figure 5.5. The process in full contains the following steps. First, large vertical and horizontal lines are removed from the image to ensure that only the textual content is present in the image. Then some image processing is done to ensure that the font does not have an effect to the whitespace calculation. This contains for example making letters I and 1 thicker as they are often abnormally located. After this all the whitespaces are calculated from the image. With the whitespaces, possible locations for spaces are calculated with kernel density estimation. When the possible spaces are located from the image, the image is split based on these points into individual words. This process is done for each image to ensure that each detection contains only a single word and not any

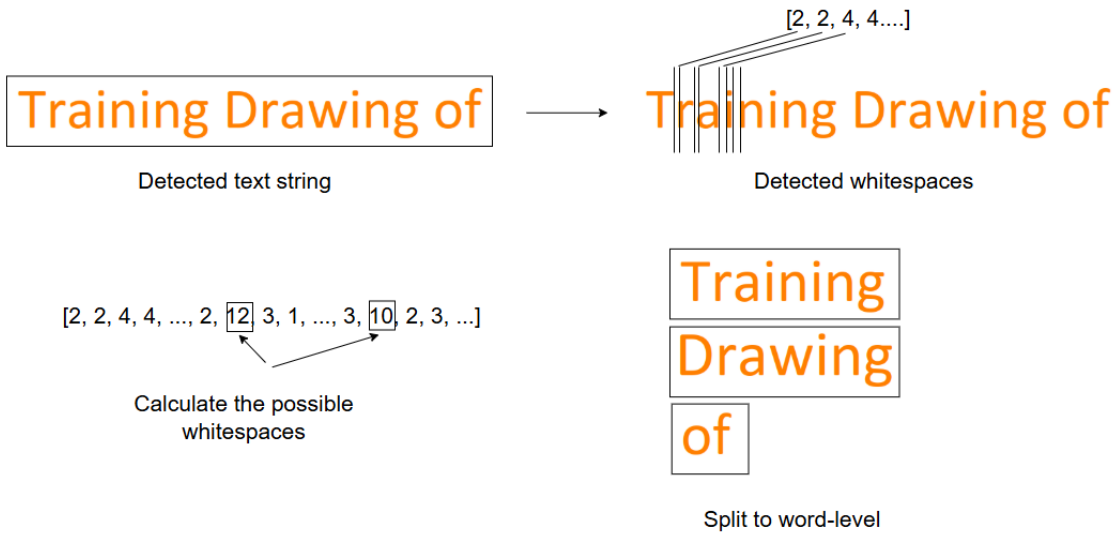

Figure 5.5: How sentences are split to words

sentences. It is important to note that the large lines, dilating, and other processing done to the image is only used for querying the whitespaces. The to-be split image is the original detection that will be also used in the recognition. After intermediate processing the resulting images are fed into the text-recognition model that utilizes SVTR-HGNet.

The varying font sizes in engineering documents were negatively affecting the word-level detections, which were needed for eShare. This whitespace splitting ensured that the recognition model was receiving only a single word at a time and could not mistakenly combine two separate words into one, which it did in the previous solutions. An example of how a single detected line of text is split into word-level images for the recognition model is visible in Figure 5.5.

For post-processing the same lexical filters were utilized that was used in the previous solutions. In addition to these there were some solutions added. The utilized PP-OCRv5 recognition library is based on the English language, but can also detect different languages, such as Chinese. As these languages contain similar characters to the English alphabet, there were incorrect detections for some characters. A step

was added which replaced all similar characters from non-English languages into the English equivalent character. This way possible mistakes in the language was fixed. An example of this was the detection of mathematical formulas in which for example 10x10 was detected so that the "x" was a multiplication x. Such changes do not negatively affect the accuracy as they are values that eShare expects.

Over the whole pipeline batching and concurrency can be utilized to speed up the processing. The intermediate and post-processing steps are easily ran concurrently without causing larger overhead and thus the processing is sped up. Also, the text recognition can be done in patches so that the detection can be marginally sped up. The whole pipeline is visible in Figure 5.6

Environment

As stated in the results in Chapter 5.2, the OCR systems cannot perform real-time detection and thus a solution is required for when to run the OCR. The proposed solution for this is to configure it so that the application will execute the OCR when the application is not being used, for example during the night. The results are then stored into an SQL database, from which they can be queried quickly when opening a document. This will cut down the processing time down upon document open in cases where the document has been already OCRd once. However, if need be, a user can OCR the document on open, which will then result in longer processing, but only for that singular open. In addition to the duration requirements, eShare is primarily developed in .net with C#, which are not widely supported by the required processing. Thus, the solution will be developed with Python and integrated into a container, which enables the possibility to run multiple concurrent containers and this way speeding up the processing, if the host machine has enough resources to do so.

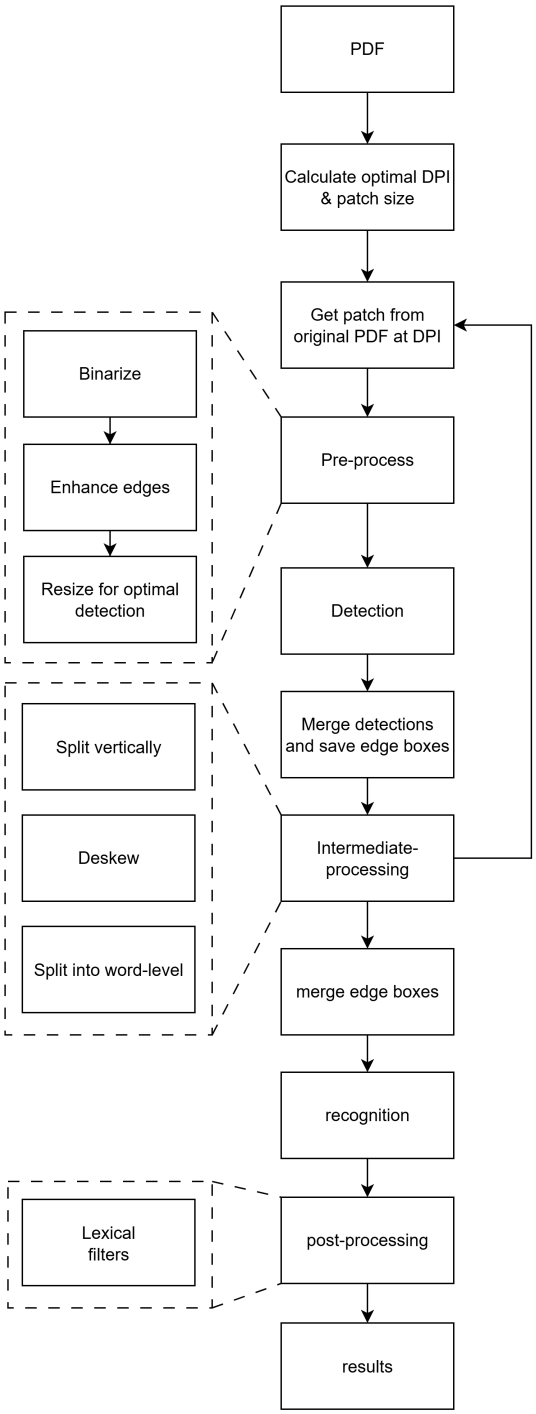

Figure 5.6: Pipeline proposed for eShare

Table 5.5: Results from the tests

Pipeline	Recall	Precision	F1
PP-HGNetV2 & SVTR-HGNet (S6)	0.839	0.787	0.812
Proposed	0.903	0.867	0.885

5.5 Solution evaluation

The previously described solution will be evaluated on the same test material and based on the same test scenario that the previous solutions were in Section 5.1. As the PP-OCRv5 solution was the best of the previous solutions, it will be used as the benchmark for the testing.

The results from the tests for the proposed solution is presented in Table 5.5 and visualized in Figure 5.7. The additional processing was able to enhance the results by a small margin. Compared to the previous solution, it is visible that the own solution exceeds the previous solution by around 8% in the F1 accuracy. The difference is small, but when approaching closer to 100%, increasing the accuracy becomes more cumbersome. Larger documents in the test set contained around 2000 text strings, which accounts to around 140 text string increase in accuracy per document, which is great when comparing the additional computational resources that this solution requires. The 4 second addition to process time is minimal and justifies the 8% increase in accuracy for it.

The proposed solution was able to exceed the accuracy of the prebuilt PP-OCRv5 pipeline and proved to be a useful solution for eShare's use-case. The PP-OCRv5 pipeline was making mistakes in the word-level detection, where it was combining detections together that were meant to be separate. The horizontal and vertical splitting that was done on the image at the intermediate processing handled this issue and provided some extra accuracy to the pipeline. The removal of unnecessary steps in the PP-OCRv5 also allowed the pipeline to function slightly faster and made room for the additional processing that was done with the custom pipeline. For ex-

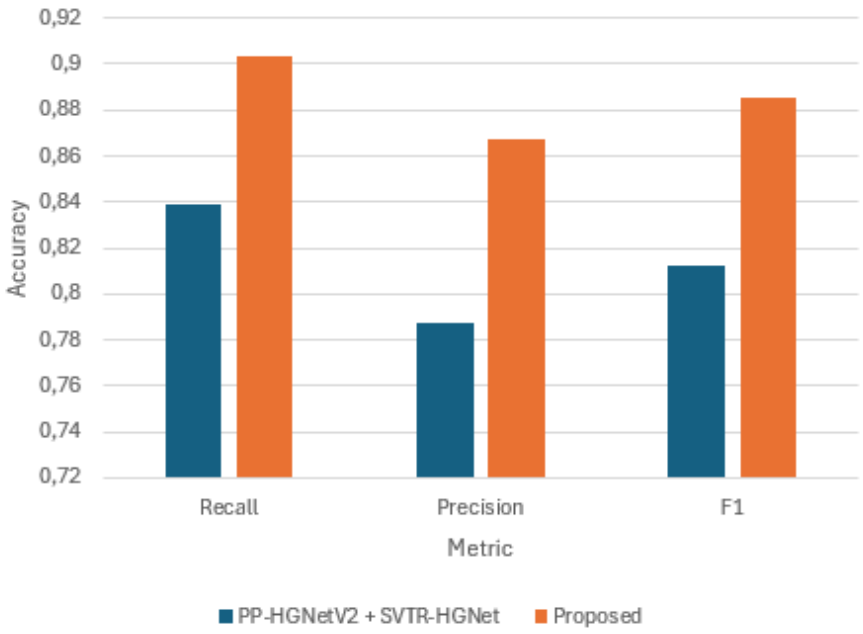

Figure 5.7: Results from the tests visualized

ample, instead of using the heavy machine learning-based orientation detection in PP-OCRv5 a lighter algorithmic deskewing function was utilized, which provided results as good as in the controlled environment of engineering drawings. As engineering documents are often in similar formats the word-level splitting was able to be done by traditional processing. This means that often after deskewing the text is vertically aligned on a white background. Thus, it is possible to calculate the whitespaces found from the image and from that algorithmically calculate the possible locations for spaces. Engineering drawings however often inhabit irregular spacing and varying fonts where the spacing might be non-traditional. Even with these issues the gain was apparent compared to when the splitting was not done. It is important to note that most of the gains were because engineering drawings are similar in format. These intermediate processing steps might not work in natural scenes which are more unpredictable.

Even though the solution is slightly slower than the PP-OCRv5, neither of them were able to provide real-time detection and thus it is quick enough for the solution.

Table 5.6: Duration of systems in seconds.

Pipeline	Small	Medium	Large	Average
PP-HGNetV2 & SVTR-HGNet (S6)	6.811	13.348	46.799	19.746
Proposed	7.977	15.704	56.743	23.771

The results for duration are visible in Table 5.6 and visualized in Figure 5.8. It is visible that the proposed solution is slightly slower than PP-OCRv5, which is to be expected from the additional processing. This addition in processing is however minimal and thus does not pose any issues for eShare's use-case as real-time detection was not possible. When the data from the OCR is stored into a cache this slowness becomes irrelevant as the document must be opened with the OCR only once and after that it can be instantly loaded from the database. The architecture in which the document is saved into the database from which it is then loaded on view merges excellently into eShare's document indexing and makes it possible for the OCR to be ran in the background. This way the user never has to wait for the document to open via the OCR but can always just load the data from the database. When documents are saved into the database it also allows the opportunity to utilize both metadata extraction and OCR in cases where the document contains some textual metadata, but not all of it. In cases such as those the data that is saved into the document can be extracted first and then the document can be extracted with OCR. By handling collisions with the detections, they can be stored into the same database and then the whole document can be extracted with maximal accuracy. This is beneficial as most often the metadata information about text is correct and is better to use than the OCRd information that might be slightly incorrect.

However, there still were issues and pain points that could not be solved even with the additional processing. The three pain points that were discussed in Section 5.2 were all attempted to be solved. The word-level splitting was able to be solved with the previously described solution. However, in the case of text being closely located inside a circle, it still proved to be difficult for the model to detect them in

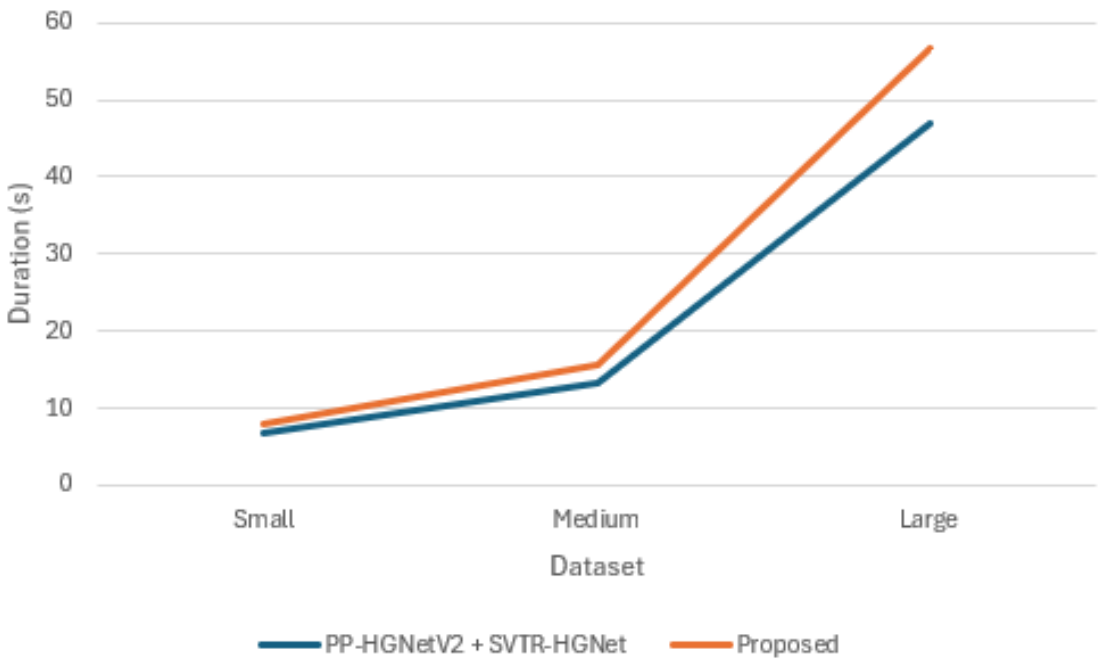

Figure 5.8: Results from the tests visualized

larger documents. It was attempted to remove all circular shapes from the image before detection, which helped in the case of smaller documents. However, with larger documents the removal was not as accurate and caused incorrect detections. The additional processing and reduced accuracy on large documents caused for the removal of circles to not be useful for this issue. In addition to the removal of circles, erosion was attempted to be utilized here as often the circles obtain thinner lines than the text. However, this also caused more incorrect detections and the benefit was minimal. Finetuning the detection model parameters and optimal DPI for the document proved to be the best solution but could not mitigate the issue completely. The detection model was having difficulties with vertically grouped text that were closely grouped. In these cases, the detection model easily grouped all the detections together. Additional processing could not help to split the text string separate from each other as there were no clear whitespaces between the lines and the recognition model could not detect multiple rows of text at the same time. Cases such as these

could possibly be generated with AI or by hand to fine-tune the detection model to handle scenarios such as these. For this thesis the finetuning of the models further is out of scope. Besides these issues, most of the incorrect detections were the result of the engineering drawing containing overlapping or otherwise invalid text strings that would be hard to detect even for a human.

Overall, the proposed solution provides an adequate solution for RQ2 and will prove useful with legacy documents. The increase in F1 accuracy compared to previous solution is significant enough to justify the pipeline instead of utilizing the prebuild PP-OCRv5 pipeline. In addition to this the pipeline gives the additional control compared to the PP-OCRv5 to modify the pipeline or even to change the detection or recognition models if wanted. Besides the previous pain points, the solution is nearly as good as a human at detecting textual strings from engineering documents.

6 Discussion

This thesis took a look into the current solution that are being used for detecting textual content from engineering drawings and proposed a solution that could be implemented into the Cadmatic's eShare information management system. With previous research it was found that there are not a lot of specialized solution for this use-case. Most of the OCR solution in engineering drawings were about symbol detection and the text detection was not as popular. This was suprising as engineering drawings inhabit alot of valuable information, which is in turn lost with legacy documents and often require manual handling by humans to extract this information. Especially for a system like eShare this is crucial. In addition to this previous solutions were able to detect with an accuracy close to 90%, which was not the case with my dataset. The dataset used contained specially difficult documents that contained hard to detect data. This most likely is caused by the time-limit imposed by the requirements, which heavily effects the accuracy of most models. For real use-cases this is crucial and cases when eShare habits thousands of documents, it would be impossible to allow the OCR to run for multiple minutes per document.

Previous solutions did not utilize a lot of processing and it was often left aside or for future research. For eShare's use-case with engineering drawings that are predictable in format, this proved to be a valuable step that enhanced the OCR with additional accuracy. EShare also contains engineering drawings for multiple fields, which made the processing difficult. Fixes to a specific field, such as P&ID,

issues would easily cause problems with other fields of engineering documents. This meant that the processing had to be as universal as possible. If the documents would have been from a single area of engineering, even more processing of the images could have been done to enhance the detection. Previous solutions also emphasized on the morphological filters to be used in pre-processing. In this thesis they did not provide any additional accuracy and often resulted in just less accuracy than the original image that was used. Eroding was useful with small images, but with larger images it caused issues and thus was not useful.

The lack of GPU usage for this solution was a clear limitor and caused issues with most of the models that were used or proposed for this thesis. For example the TrOCR model is not suitable for use with CPUs as the duration that it took to detect was too much. This is to be expected as the transformer architecture is directed towards GPUs and not optimized for CPUs. However, PaddleOCR was suprisingly optimized for CPU performance and was able to provide great results regardless of the limited computational power. PaddleOCR proved to be a suitable library for the use-case and leaves the door open for future development with the models. There is also the possiblity to fine-tune the models in case a customer wants a customized model or data for training is optimized from somewhere.

Engineering drawings also proved to be difficult to process even by hand. Many engineering drawings were drawn with confusing fonts that left room open for discussion on what specific characters were and where some sentences should split into words or if they were to be combined into a single string. This caused issues in creating the dataset and making it as nonbiased as possible, which in turn caused some missed detections even though the detection functioned as expected. To enhance the dataset for testing, multiple persons should have went through it, but that was not a possiblity in this thesis. This causes the results slightly skewing and not maybe being the most accurate. However, the large amount of data in the dataset should

offset this and give accuracy. In some cases it would be impossible to definitely say if two words should be seperate or combined, which is caused by the fonts used in the documents.

The thesis offered a solution that provided satisfactory results for implementation into eShare. It is not the optimal solution, as real-time detection and better accuracy would have been preferred, but compared to previous solutions and the use-case, the solution proposed is satisfactory. Real-time detection is possibly achievable even with CPUs, but only with smaller documents, such as A4. For larger A0 documents, and especially with documents that contained a lot of data, the recognition became an issue. To have access to GPUs, real-time detection could possibly be also implemented for large A0 size engineering documents. As eShare did not have any extraction of text from legacy engineers previously, this solution provides a lot of value. As the OCR is based on a pre-existing model, it can be also utilized in other environments than engineering drawings if need be. As mentioned in Chapter 3 eShare contains a wide variety of documents in addition to engineering drawings, which might face the same issues as engineering drawings and the solution proposed here can provide benefits there, even though it is not made exactly for that environment.

Future research

This thesis explored the possiblity of enhancing results with specialized intermediate processing, but this research could be taken even further. For future research I propose the three following research questions:

- Can AI be utilized to create an artificial engineering drawing dataset?
- Can specific pain points of engineering document detection be solved with finetuning?
- How can specialized intermediate processing be utilized to enhance OCR for

specific types of engineering documents?

The first question, handling on the issue of the lack of engineering drawings that could be utilized for fine-tuning models. There is a possibility of utilizing AI for generating artificial engineering drawings that could be utilized for finetuning models. I do not believe that existing models could do this out-of-the-box, but creating a specified model could prove extreamly useful here. On a lower scope, the second question handles the pain points that were brought up in this thesis and were not able to be solved. With AI or by hand a dataset could be created that contains these pain points and by finetuning a model could improve the rate of detections. Finally, there were ideas in this thesis for solving the pain points, but most of them did not work as the use-case for eShare contains a range of different kinds of documents. However, in a case where there was only a single field of engineering documents, a more specialized intermediate processing and pre-processing could enhance the detection even further. Answering these two questions would need additional research that is out of the scope of this thesis, but would provide useful information in the area of optical character recognition of engineering documents.

7 Conclusion

Engineering drawings inhabit alot of textual information that is valuable to the user and to eShare as a system to enable functionality, such as document-model linking. With legacy documents or in cases where the conversion to PDFs the textual data is lost from the metadata of the document and it can not be extracted using traditional methods. In such cases a human would have to manually go and extract the data, which is slow and cumbersome to the user.

This thesis was done to address the issue of extracting textual information from engineering documents. Two research questions were provided:

RQ1: What are the current solutions for detecting part numbers from engineering drawings and how do they compare?

RQ2: What is the best implementation for eShare and how can this be implemented?

The first question, handles the research of previous solutions that are currently being used for this use-case. It was addressed in Chapter 4. Multiple solutions were found, which addressed the issue at hand. However, these solutions were not directly what was required for eShare's use-case and thus they were evaluated for the specified use-case in Chapter 5, which addresses the second half of the question. Previous solutions, which were old and had newer and more performative solutions, were upgraded and additional solution were provided. The most optimal solution from the previous solutions was the PP-OCRv5 pipeline that is created by PaddleOCR.

With the information gathered from the RQ1, the RQ2 was addressed in Chapter 5. The chapter contained a proposed solution for eShare, which was then in turn benchmarked compared to the best previous solution, PP-OCRv5. The solution implemented enhanced intermediate processing, which was lacking in the previous pipelines. It was able to prove a 8% increase to the previous pipeline, with additional intermediate processing that was focused on engineering drawings and especially on splitting detections to word-level for the recognition model. In total the proposed pipeline was able to achieve a recall rate of 89% and an F1 score of 81%.

The thesis was able to find a suitable implementation from the previous solutions, but also discovered that the previous solutions were not directly applicable to eShare's unique use-case. With this information, a pipeline was proposed that fulfilled the needs for eShare and proved satisfactory results, which can enhance the experience and add value to eShare.

References

[1] C. F. Moreno-García, E. Elyan, and C. Jayne, "New trends on digitisation of complex engineering drawings", en, *Neural Computing and Applications*, vol. 31, no. 6, pp. 1695–1712, Jun. 2019, ISSN: 0941-0643, 1433-3058. DOI: 10.1007/s00521-018-3583-1. Accessed: May 9, 2025. [Online]. Available: http://link.springer.com/10.1007/s00521-018-3583-1.

[2] C. Maupou et al., "Automatic raster engineering drawing digitisation for legacy parts towards advanced manufacturing", en, *Procedia CIRP*, vol. 129, pp. 234–239, 2024, ISSN: 22128271. DOI: 10.1016/j.procir.2024.10.041. Accessed: May 9, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212827124011776.

[3] ISO, *ISO 128-1:2020(en), Technical product documentation (TPD) — General principles of representation — Part 1: Introduction and fundamental requirements*. Accessed: May 9, 2025. [Online]. Available: https://www.iso.org/obp/ui/en/#iso:std:iso:128:-1:ed-2:v1:en.

[4] McGill University. "Working Drawings and Assemblies". en, Accessed: May 9, 2025. [Online]. Available: https://www.mcgill.ca/engineeringdesign/step-step-design-process/basics-graphics-communication/working-drawings-and-assemblies.

[5] SyBridge Technologies, *What to Include in Your Engineering Drawing*, en-US, Feb. 2022. Accessed: May 9, 2025. [Online]. Available: https://sybridge.com/what-to-include-in-engineering-drawing/.

[6] IBM, *What is Computer Vision? | IBM*. Accessed: Apr. 21, 2025. [Online]. Available: https://www.ibm.com/think/topics/computer-vision.

[7] Intel, *What Is Computer Vision?*, en. Accessed: Apr. 21, 2025. [Online]. Available: https://www.intel.com/content/www/us/en/learn/what-is-computer-vision.html.

[8] P. S. R. Kishore, "ClueNet : A Deep Framework for Occluded Pedestrian Pose Estimation", en,

[9] D. Bhatt et al., "CNN Variants for Computer Vision: History, Architecture, Application, Challenges and Future Scope", en, *Electronics*, vol. 10, no. 20, p. 2470, Oct. 2021, ISSN: 2079-9292. DOI: 10.3390/electronics10202470. Accessed: Apr. 24, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/10/20/2470.

[10] IBM, *What are Convolutional Neural Networks? | IBM*, en, Oct. 2021. Accessed: Apr. 24, 2025. [Online]. Available: https://www.ibm.com/think/topics/convolutional-neural-networks.

[11] IBM, *What is a Recurrent Neural Network (RNN)? | IBM*, en, Oct. 2021. Accessed: Apr. 24, 2025. [Online]. Available: https://www.ibm.com/think/topics/recurrent-neural-networks.

[12] B. Shi, X. Bai, and C. Yao, *An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition*, arXiv:1507.05717 [cs], Jul. 2015. DOI: 10.48550/arXiv.1507.05717. Accessed: Apr. 24, 2025. [Online]. Available: http://arxiv.org/abs/1507.05717.

[13] A. Vaswani et al., *Attention Is All You Need*, en, arXiv:1706.03762 [cs], Aug. 2023. DOI: 10.48550/arXiv.1706.03762. Accessed: May 21, 2025. [Online]. Available: http://arxiv.org/abs/1706.03762.

[14] IBM, *What is a Transformer Model? | IBM*, en, Mar. 2025. Accessed: Apr. 24, 2025. [Online]. Available: https://www.ibm.com/think/topics/transformer-model.

[15] K. N. Haque, *What is Convolutional Neural Network — CNN (Deep Learning)*, en, Feb. 2023. Accessed: Apr. 24, 2025. [Online]. Available: https://nafizshahriar.medium.com/what-is-convolutional-neural-network-cnn-deep-learning-b3921bdd82d5.

[16] IBM, *What is Fine-Tuning? | IBM*, en, Mar. 2024. Accessed: Apr. 24, 2025. [Online]. Available: https://www.ibm.com/think/topics/fine-tuning.

[17] Amazon, *What is LLM? - Large Language Models Explained - AWS*, en-US. Accessed: Apr. 24, 2025. [Online]. Available: https://aws.amazon.com/what-is/large-language-model/.

[18] A. Hannun, "Sequence Modeling with CTC", en, *Distill*, vol. 2, no. 11, e8, Nov. 2017, ISSN: 2476-0757. DOI: 10.23915/distill.00008. Accessed: May 12, 2025. [Online]. Available: https://distill.pub/2017/ctc.

[19] M. S. M. Yazed, E. F. A. Shaubari, and M. H. Yap, "A Review of Neural Network Approach on Engineering Drawing Recognition and Future Directions", en,

[20] X.-F. Wang, Z.-H. He, K. Wang, Y.-F. Wang, L. Zou, and Z.-Z. Wu, "A survey of text detection and recognition algorithms based on deep learning technology", en, *Neurocomputing*, vol. 556, p. 126 702, Nov. 2023, ISSN: 09252312. DOI: 10.1016/j.neucom.2023.126702. Accessed: Apr. 29, 2025.

[Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0925231223008251.

[21] H. Wang, C. Pan, X. Guo, C. Ji, and K. Deng, "From object detection to text detection and recognition: A brief evolution history of optical character recognition", en, *WIREs Computational Statistics*, vol. 13, no. 5, e1547, Sep. 2021, ISSN: 1939-5108, 1939-0068. DOI: 10.1002/wics.1547. Accessed: Apr. 29, 2025. [Online]. Available: https://wires.onlinelibrary.wiley.com/doi/10.1002/wics.1547.

[22] Z. Raisi, M. A. Naiel, P. Fieguth, S. Wardell, and J. Zelek, *Text Detection and Recognition in the Wild: A Review*, en, arXiv:2006.04305 [cs], Jun. 2020. DOI: 10.48550/arXiv.2006.04305. Accessed: May 6, 2025. [Online]. Available: http://arxiv.org/abs/2006.04305.

[23] X. Zhou et al., "EAST: An Efficient and Accurate Scene Text Detector", en, in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Honolulu, HI: IEEE, Jul. 2017, pp. 2642–2651, ISBN: 978-1-5386-0457-1. DOI: 10.1109/CVPR.2017.283. Accessed: May 6, 2025. [Online]. Available: http://ieeexplore.ieee.org/document/8099766/.

[24] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character Region Awareness for Text Detection", en, in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Long Beach, CA, USA: IEEE, Jun. 2019, pp. 9357–9366, ISBN: 978-1-72813-293-8. DOI: 10.1109/CVPR.2019.00959. Accessed: May 6, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/8953846/.

[25] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, *Real-time Scene Text Detection with Differentiable Binarization*, en, arXiv:1911.08947 [cs], Dec. 2019. DOI:

10.48550/arXiv.1911.08947. Accessed: May 14, 2025. [Online]. Available: http://arxiv.org/abs/1911.08947.

[26] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks", en, in *Proceedings of the 23rd international conference on Machine learning - ICML '06*, Pittsburgh, Pennsylvania: ACM Press, 2006, pp. 369–376, ISBN: 978-1-59593-383-6. DOI: 10.1145/1143844.1143891. Accessed: Aug. 24, 2025. [Online]. Available: http://portal.acm.org/citation.cfm?doid=1143844.1143891.

[27] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[28] F. Morales, *Faustomorales/keras-ocr*, original-date: 2019-09-20T23:08:50Z, Jun. 2025. Accessed: Jun. 12, 2025. [Online]. Available: https://github.com/faustomorales/keras-ocr.

[29] Jaided AI, *JaidedAI/EasyOCR*, original-date: 2020-03-14T11:46:39Z, Jun. 2025. Accessed: Jun. 12, 2025. [Online]. Available: https://github.com/JaidedAI/EasyOCR.

[30] Paddle OCR, *Overview - PaddleOCR Documentation*. Accessed: May 13, 2025. [Online]. Available: https://paddlepaddle.github.io/PaddleOCR/main/en/ppocr/overview.html.

[31] R. Smith, "An Overview of the Tesseract OCR Engine", in *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, ISSN: 2379-2140, vol. 2, Sep. 2007, pp. 629–633. DOI: 10.1109/ICDAR.2007.4376991. Accessed: Jun. 12, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/4376991/.

[32] Tesseract, *Tesseract User Manual*, en-US. Accessed: May 6, 2025. [Online]. Available: https://tesseract-ocr.github.io/tessdoc/.

[33] M. Li et al., "TrOCR: Transformer-Based Optical Character Recognition with Pre-trained Models", en, *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 11, pp. 13 094–13 102, Jun. 2023, ISSN: 2374-3468, 2159-5399. DOI: 10.1609/aaai.v37i11.26538. Accessed: May 6, 2025. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/26538.

[34] Y. Du et al., *SVTR: Scene Text Recognition with a Single Visual Model*, en, arXiv:2205.00159 [cs], May 2022. DOI: 10.48550/arXiv.2205.00159. Accessed: May 8, 2025. [Online]. Available: http://arxiv.org/abs/2205.00159.

[35] R. Hernández-García, R. J. Barrientos, and S. A. Velastin, Eds., *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications: 27th Iberoamerican Congress, CIARP 2024, Talca, Chile, November 26–29, 2024, Proceedings, Part I* (Lecture Notes in Computer Science), en. Cham: Springer Nature Switzerland, 2025, vol. 15368, ISBN: 978-3-031-76606-0 978-3-031-76607-7. DOI: 10.1007/978-3-031-76607-7. Accessed: May 21, 2025. [Online]. Available: https://link.springer.com/10.1007/978-3-031-76607-7.

[36] N. Girdhar, M. Coustaty, and A. Doucet, "Digitizing History: Transitioning Historical Paper Documents to Digital Content for Information Retrieval and Mining—A Comprehensive Survey", en, *IEEE Transactions on Computational Social Systems*, vol. 11, no. 5, pp. 6151–6180, Oct. 2024, ISSN: 2329-924X, 2373-7476. DOI: 10.1109/TCSS.2024.3378419. Accessed: May 21, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10495892/.

[37] A. Alaei, V. Bui, D. Doermann, and U. Pal, "Document Image Quality Assessment: A Survey", en, *ACM Computing Surveys*, vol. 56, no. 2, pp. 1–36, Feb. 2024, ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3606692. Accessed: May 21, 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/3606692.

[38] R. Dey, R. C. Balabantaray, S. Mohanty, D. Singh, M. Karuppiah, and D. Samanta, "Approach for Preprocessing in Offline Optical Character Recognition (OCR)", en, in *2022 Interdisciplinary Research in Technology and Management (IRTM)*, Kolkata, India: IEEE, Feb. 2022, pp. 1–6, ISBN: 978-1-66547-886-1. DOI: 10.1109/IRTM54583.2022.9791698. Accessed: May 7, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9791698/.

[39] I. Hammad, M. De Costa, A. Boroomand, and M. Anwar, "Automating equipment identification in nuclear engineering drawings", en, *Nuclear Engineering and Design*, vol. 436, p. 114 002, May 2025, ISSN: 00295493. DOI: 10.1016/j.nucengdes.2025.114002. Accessed: May 7, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0029549325001797.

[40] T. T. H. Nguyen, A. Jatowt, M. Coustaty, and A. Doucet, "Survey of Post-OCR Processing Approaches", en, *ACM Computing Surveys*, vol. 54, no. 6, pp. 1–37, Jul. 2022, ISSN: 0360-0300, 1557-7341. DOI: 10.1145/3453476. Accessed: May 7, 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/3453476.

[41] A. Choudhary, R. Rishi, and S. Ahlawat, "A New Character Segmentation Approach for Off-Line Cursive Handwritten Words", en, *Procedia Computer Science*, vol. 17, pp. 88–95, 2013, ISSN: 18770509. DOI: 10.1016/j.procs.2013.05.013. Accessed: May 7, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1877050913001464.

[42] F. J. Damerau, "A technique for computer detection and correction of spelling errors", en, *Communications of the ACM*, vol. 7, no. 3, pp. 171–176, Mar. 1964, ISSN: 0001-0782, 1557-7317. DOI: 10.1145/363958.363994. Accessed: May 21, 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/363958.363994.

[43] J. Kanerva, C. Ledins, S. Käpyaho, and F. Ginter, *OCR Error Post-Correction with LLMs in Historical Documents: No Free Lunches*, arXiv:2502.01205 [cs], Feb. 2025. DOI: 10.48550/arXiv.2502.01205. Accessed: Aug. 27, 2025. [Online]. Available: http://arxiv.org/abs/2502.01205.

[44] Y. Ren, H. Yao, G. Liu, and Z. Bai, "A Text Code Recognition and Positioning System for Engineering Drawings of Nuclear Power Equipment", en, in *2022 IEEE 6th Information Technology and Mechatronics Engineering Conference (ITOEC)*, Chongqing, China: IEEE, Mar. 2022, pp. 661–665, ISBN: 978-1-66543-185-9. DOI: 10.1109/ITOEC53115.2022.9734621. Accessed: May 7, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9734621/.

[45] S. Uchida, E. Barney, and V. Eglin, Eds., *Document Analysis Systems: 15th IAPR International Workshop, DAS 2022, La Rochelle, France, May 22–25, 2022, Proceedings* (Lecture Notes in Computer Science), en. Cham: Springer International Publishing, 2022, vol. 13237, ISBN: 978-3-031-06554-5 978-3-031-06555-2. DOI: 10.1007/978-3-031-06555-2. Accessed: May 7, 2025. [Online]. Available: https://link.springer.com/10.1007/978-3-031-06555-2.

[46] W. Khallouli, M. S. Uddin, A. Sousa-Poza, J. Li, and S. Kovacic, "Leveraging Transformer-Based OCR Model with Generative Data Augmentation for Engineering Document Recognition", en, *Electronics*, vol. 14, no. 1, p. 5, Dec. 2024, ISSN: 2079-9292. DOI: 10.3390/electronics14010005. Accessed: May 7, 2025. [Online]. Available: https://www.mdpi.com/2079-9292/14/1/5.

[47] V. Shteriyanov, R. Dzhusupova, J. Bosch, and H. Holmström Olsson, "Artificial intelligence-based solution for the automatic extraction of pipeline metadata from piping and instrumentation diagrams (p&ids)", *Available at SSRN 5113430*,

[48] W. Wang et al., *Shape Robust Text Detection with Progressive Scale Expansion Network*, en, arXiv:1903.12473 [cs], Jul. 2019. DOI: 10.48550/arXiv.1903.12473. Accessed: May 8, 2025. [Online]. Available: http://arxiv.org/abs/1903.12473.

[49] PaddleOCR, *PP-OCRv5 Introduction - PaddleOCR Documentation*. Accessed: Aug. 11, 2025. [Online]. Available: https://www.paddleocr.ai/latest/en/version3.x/algorithm/PP-OCRv5/PP-OCRv5.html.

[50] PaddleOCR, *Usage Tutorial - PaddleOCR Documentation*. Accessed: May 23, 2025. [Online]. Available: https://paddlepaddle.github.io/PaddleOCR/main/en/version3.x/pipeline_usage/OCR.html#1-ocr-pipeline-introduction.

[51] C. Li et al., *PP-OCRv3: More Attempts for the Improvement of Ultra Lightweight OCR System*, en, arXiv:2206.03001 [cs], Jun. 2022. DOI: 10.48550/arXiv.2206.03001. Accessed: May 23, 2025. [Online]. Available: http://arxiv.org/abs/2206.03001.

[52] C. Cui et al., *PP-LCNet: A Lightweight CPU Convolutional Neural Network*, en, arXiv:2109.15099 [cs], Sep. 2021. DOI: 10.48550/arXiv.2109.15099. Accessed: May 23, 2025. [Online]. Available: http://arxiv.org/abs/2109.15099.

[53] Google, *Classification: Accuracy, recall, precision, and related metrics | Machine Learning*, en. Accessed: May 28, 2025. [Online]. Available: https://

developers.google.com/machine-learning/crash-course/classification/accuracy-precision-recall.

[54] M. M. Agritania and M. M. Isnaini, "Development of an engineering drawing detection and extraction algorithm for quality inspection using deep neural networks", en, *Procedia CIRP*, vol. 132, pp. 135–140, 2025, ISSN: 22128271. DOI: 10.1016/j.procir.2025.01.023. Accessed: Aug. 26, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S221282712500023X.

[55] R. Avyodri, S. Lukas, and H. Tjahyadi, "Optical Character Recognition (OCR) for Text Recognition and its Post-Processing Method: A Literature Review", en, in *2022 1st International Conference on Technology Innovation and Its Applications (ICTIIA)*, Tangerang, Indonesia: IEEE, Sep. 2022, pp. 1–6, ISBN: 978-1-66548-826-6. DOI: 10.1109/ICTIIA54654.2022.9935961. Accessed: May 6, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9935961/.

[56] Q. A. Bui, D. Mollard, and S. Tabbone, "Selecting Automatically Pre-Processing Methods to Improve OCR Performances", en, in *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Kyoto: IEEE, Nov. 2017, pp. 169–174, ISBN: 978-1-5386-3586-5. DOI: 10.1109/ICDAR.2017.36. Accessed: Aug. 26, 2025. [Online]. Available: http://ieeexplore.ieee.org/document/8269967/.

[57] I. Campiotti and R. Lotufo, "Optical character recognition with transformers and CTC", en, in *Proceedings of the 22nd ACM Symposium on Document Engineering*, San Jose California: ACM, Sep. 2022, pp. 1–4, ISBN: 978-1-4503-9544-1. DOI: 10.1145/3558100.3563845. Accessed: Aug. 26, 2025. [Online]. Available: https://dl.acm.org/doi/10.1145/3558100.3563845.

[58] S. M. Gajbhiye, S. R. Bhamre, L. N. T. Tadepalli, M. R. Pillai, and D. Uplaonkar, "Advancing P&ID Digitization with YOLOv5", en, in *2023 International Conference on Integrated Intelligence and Communication Systems (ICIICS)*, Kalaburagi, India: IEEE, Nov. 2023, pp. 1–6, ISBN: 9798350315455. DOI: 10.1109/ICIICS59993.2023.10421368. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10421368/.

[59] H. Gonuguntla, K. Meghana, T. S. Prajwal, K. Nvss, K. N. Sai, and C. Jain, "Evaluating Modern Information Extraction Techniques for Complex Document Structures", en, in *2024 International Conference on Electrical, Computer and Energy Technologies (ICECET*, Sydney, Australia: IEEE, Jul. 2024, pp. 1–5, ISBN: 9798350395914. DOI: 10.1109/ICECET61485.2024.10698618. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10698618/.

[60] Y. He, "Research on Text Detection and Recognition Based on OCR Recognition Technology", en, in *2020 IEEE 3rd International Conference on Information Systems and Computer Aided Education (ICISCAE)*, Dalian, China: IEEE, Sep. 2020, pp. 132–140, ISBN: 978-1-72818-304-6. DOI: 10.1109/ICISCAE51034.2020.9236870. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/9236870/.

[61] A. Kashevnik et al., "An Approach to Engineering Drawing Organization: Title Block Detection and Processing", en, *IEEE Access*, pp. 1–1, 2024, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3244603. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10043657/.

[62] Y.-H. Lin, Y.-H. Ting, Y.-C. Huang, K.-L. Cheng, and W.-R. Jong, "Integration of Deep Learning for Automatic Recognition of 2D Engineering Drawings", en, *Machines*, vol. 11, no. 8, p. 802, Aug. 2023, ISSN: 2075-1702. DOI:

10.3390/machines11080802. Accessed: Aug. 26, 2025. [Online]. Available: https://www.mdpi.com/2075-1702/11/8/802.

[63] "Intelligent Reading for Multi-Scale Engineering Drawings based on Adaptive Object Detection and Optical Character Recognition", en, in *Proceedings of WCSE 2022 Spring Event: 2022 9th International Conference on Industrial Engineering and Applications*, WCSE, 2022, ISBN: 978-981-18585-2-9. DOI: 10.18178/wcse.2022.04.171. Accessed: Aug. 26, 2025. [Online]. Available: http://www.wcse.org/WCSE%202022%20Spring/171.pdf.

[64] A. R. Putra, S. Ha, and K.-P. Park, "Automatic extraction of cable connection information from 2D drawings for electrical outfittings design in shipyards", en, *International Journal of Naval Architecture and Ocean Engineering*, vol. 16, p. 100 630, 2024, ISSN: 20926782. DOI: 10.1016/j.ijnaoe.2024.100630. Accessed: Aug. 26, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2092678224000499.

[65] R. Rahul, S. Paliwal, M. Sharma, and L. Vig, *Automatic Information Extraction from Piping and Instrumentation Diagrams*, en, arXiv:1901.11383 [cs], Jan. 2019. DOI: 10.48550/arXiv.1901.11383. Accessed: Aug. 26, 2025. [Online]. Available: http://arxiv.org/abs/1901.11383.

[66] B. Shi, X. Bai, and C. Yao, *An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition*, en, arXiv:1507.05717 [cs], Jul. 2015. DOI: 10.48550/arXiv.1507.05717. Accessed: Aug. 26, 2025. [Online]. Available: http://arxiv.org/abs/1507.05717.

[67] V. Shteriyanov, R. Dzhusupova, J. Bosch, and H. Holmström Olsson, "Robust Detection of Line Numbers in Piping and Instrumentation Diagrams (P&IDs)", en, in *2024 International Conference on Machine Learning and Ap-*

plications (ICMLA), Miami, FL, USA: IEEE, Dec. 2024, pp. 888–893, ISBN: 9798350374889. DOI: 10.1109/ICMLA61862.2024.00129. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10903343/.

[68] J. V. Toro and M. Tarkian, "Optimizing Text Recognition in Mechanical Drawings: A Comprehensive Approach", en, *Machines*, vol. 13, no. 3, p. 254, Mar. 2025, ISSN: 2075-1702. DOI: 10.3390/machines13030254. Accessed: Aug. 26, 2025. [Online]. Available: https://www.mdpi.com/2075-1702/13/3/254.

[69] B. Wang, Y. Cui, C. Zhang, M. Xie, D. Wang, and D. Chen, "Image Recognition Empowers Intelligent Analysis of Industrial Drawings", en, in *2024 6th International Conference on Industrial Artificial Intelligence (IAI)*, Shenyang, China: IEEE, Aug. 2024, pp. 1–6, ISBN: 9798350356618. DOI: 10.1109/IAI63275.2024.10730735. Accessed: Aug. 26, 2025. [Online]. Available: https://ieeexplore.ieee.org/document/10730735/.

[70] M. T. Khan, Z. Yong, L. Chen, J. M. Tan, W. Feng, and S. K. Moon, "Automated Parsing of Engineering Drawings for Structured Information Extraction Using a Fine-tuned Document Understanding Transformer", en,

[71] T. Schlagenhauf, M. Netzer, and J. Hillinger, "Text Detection on Technical Drawings for the Digitization of Brown-field Processes", en, *Procedia CIRP*, vol. 118, pp. 372–377, 2023, ISSN: 22128271. DOI: 10.1016/j.procir.2023.06.064. Accessed: Aug. 26, 2025. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S2212827123002883.

Appendix A

Table A.1: Read articles for previous implementations.

Number	Author	Title
1	M. M. Agritania et al. [54]	Development of an engineering drawing detection and extraction algorithm for quality inspection using deep neural networks
2	R. Avyodri [55]	Optical Character Recognition (OCR) for Text Recognition and its Post-Processing Method: A Literature Review
3	Y. Baek et al. [24]	Character Region Awareness for Text Detection
4	Q. A. Bui et al. [56]	Selecting Automatically Pre-Processing Methods to Improve OCR Performances
5	I. Campiotti et al. [57]	Optical character recognition with transformers and CTC
6	A. Choudhary et al. [41]	A New Character Segmentation Approach for Off-Line Cursive Handwritten Words
7	R. Dey et al. [38]	Approach for Preprocessing in Offline Optical Character Recognition (OCR)

8	S. M. Gajbhiye et al. [58]	Advancing PID Digitization with YOLOv5
9	H. Gonuguntla et al. [59]	Evaluating Modern Information Extraction Techniques for Complex Document Structures
10	I. Hammad et al. [39]	Automating equipment identification in nuclear engineering drawings
11	Y. He [60]	Research on Text Detection and Recognition Based on OCR Recognition Technology
12	A. Kashevnik et al. [61]	An Approach to Engineering Drawing Organization: Title Block Detection and Processing
13	W. Khallouli et al. [46]	Leveraging Transformer-Based OCR Model with Generative Data Augmentation for Engineering Document Recognition
14	Y.-H. Lin et al. [62]	Integration of Deep Learning for Automatic Recognition of 2D Engineering Drawings
15	C. Maupou [2]	Automatic raster engineering drawing digitisation for legacy parts towards advanced manufacturing
16	C. F. Moreno-García et al. [1]	New trends on digitisation of complex engineering drawings
17	T. T. H. Nguyen et al. [40]	Survey of Post-OCR Processing Approaches

18	R. Yao et al. [63]	Intelligent Reading for Multi-Scale Engineering Drawings based on Adaptive Object Detection and Optical Character Recognition
19	A. R. Putra et al. [64]	Automatic extraction of cable connection information from 2D drawings for electrical outfittings design in shipyards
20	R. Rahul et al. [65]	Automatic Information Extraction from Piping and Instrumentation Diagrams
21	Z. Raisi et al. [22]	Text Detection and Recognition in the Wild: A Review
22	Y. Ren et al. [44]	A Text Code Recognition and Positioning System for Engineering Drawings of Nuclear Power Equipment
23	B. Shi et al. [66]	An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition
24	V. Shteriyanov et al. [67]	Robust Detection of Line Numbers in Piping and Instrumentation Diagrams (PIDs)
25	J. V. Toro et al. [68]	Optimizing Text Recognition in Mechanical Drawings: A Comprehensive Approach
26	B. Wang et al. [69]	Image Recognition Empowers Intelligent Analysis of Industrial Drawings
27	H. Wang et al. [21]	From object detection to text detection and recognition: A brief evolution history of optical character recognition

28	W. Wang et al. [48]	Shape Robust Text Detection with Progressive Scale Expansion Network
29	X.-F. Wang et al. [20]	A survey of text detection and recognition algorithms based on deep learning technology
30	Francois et al. [45]	Text Detection and Post-OCR Correction in Engineering Documents
31	A. Kashevnik et al. [61]	An Approach to Engineering Drawing Organization: Title Block Detection and Processing
32	M. T. Khan et al. [70]	Automated Parsing of Engineering Drawings for Structured Information Extraction Using a Fine-tuned Document Understanding Transformer
33	T. Schlagenhauf et al. [71]	Text Detection on Technical Drawings for the Digitization of Brown-field Processes
34	M. S. M. Yazed et al. [19]	A Review of Neural Network Approach on Engineering Drawing Recognition and Future Directions