Computers in Industry 129 (2021) 103442

Contents lists available at ScienceDirect

Computers in Industry

journal homepage: www.elsevier.com/locate/compind

Extraction of dimension requirements from engineering drawings for supporting quality control in production processes

Beate Scheibel[a,*], Juergen Mangler[b], Stefanie Rinderle-Ma[b]

[a] Research Group Workflow Systems and Technology, Faculty of Computer Science, University of Vienna, Waehringerstrasse 29, 1090 Vienna, Austria
[b] Information Systems and Business Process Management, Department of Informatics, Technical University of Munich, Boltzmannstrasse 3, 85748 Garching, Germany

ARTICLE INFO

Article history:
Received 6 October 2020
Received in revised form 5 March 2021
Accepted 10 March 2021
Available online 24 March 2021

Keywords:
Engineering drawings
Information extraction
Clustering
Computer aided quality control
Manufacturing process quality

ABSTRACT

Engineering drawings accompany a workpiece throughout its production process and include information about the dimensions and tolerances as well as the associated regulatory standards. Even though the construction and manufacturing process of a workpiece can be almost entirely performed automatically, the design and use of engineering drawings is still not fully integrated in the automated production process. This work provides DigiEDraw, a conceptual approach as well as a prototype to extract dimensioning information from engineering drawings and to integrate this information into the production process to facilitate and optimize quality control. The extraction process is based on 2D clustering. The challenge is to determine the parameters to distinguish clusters representing different dimensioning information. The approach uses DBSCAN, achieving a recall value of over 88%. The applicability of DigiEDraw is demonstrated based on a real-world manufacturing process.

1. Introduction

Production processes usually require a model of the workpiece as input. Typically, these models are designed using a *computer-aided design (CAD)* program such as AutoCAD or Solidworks. The model is then transformed into a numerical control (NC) program using computer-aided manufacturing (CAM) tools, for example, Catio or Esprit. The NC program, in turn, is used by a tooling machine to manufacture the workpiece. Design and production processes can be seen as two separate steps or can be developed together using an integrated CAD-CAM system.

Engineering drawings (EDs) are 2D depictions of a workpiece that include geometric as well as textual information such as measurements, tolerances, and applicable norms, which are essential for *quality control* of the finished workpiece. CAD modeling describes the design of a workpiece with the help of CAD programs. Hence, EDs are strictly speaking CAD models. However, this paper uses the term CAD model for a digital model of a workpiece, usually in 3D, that only includes graphical and geometrical information, whereas the term EDs refers to manual and digital drawings that include 2D depictions of a workpiece as well as dimensioning information. Examples of a 3D model of a workpiece, an ED, and the resulting workpiece can be seen in Fig. 1. EDs can be generated from a CAD model. However, additional information (i.e., tolerances and standards) has to be added manually as it is not included in CAD models by default.

Nowadays, CAD models are typically used for the actual production process. Nevertheless, EDs are still mostly applied as the contractual basis and as reference for quality control as the specifications of tolerances as well as the applicable standards are essential for these purposes (Labisch and Weber, 2008). According to Henderson (2014), 250 million new EDs are generated each year and millions of legacy EDs are still in circulation. A solution that allows the extraction of information from EDs, that is not included in the CAD model, can be used to automate the entire production process including measuring and quality control. An optimal solution should be able to extract all data including *graphical elements* as well as additional information such as the *dimensioning requirements*. However, it is not always necessary to extract geometric and graphical elements, as an additional CAD model exists in a lot of cases. The problem is to include *dimensioning information* in the process to create a seamless production chain. This refers not only to the *dimensions* and *tolerances* written on the ED itself, but also to the information that is part of an associated *regulatory framework*, e.g., ISO or DIN standards. These regulatory documents usually specify minimum standards that should be sat-

* Corresponding author.
E-mail addresses: beate.scheibel@univie.ac.at (B. Scheibel), juergen.mangler@tum.de (J. Mangler), stefanie.rinderle-ma@tum.de (S. Rinderle-Ma).

https://doi.org/10.1016/j.compind.2021.103442

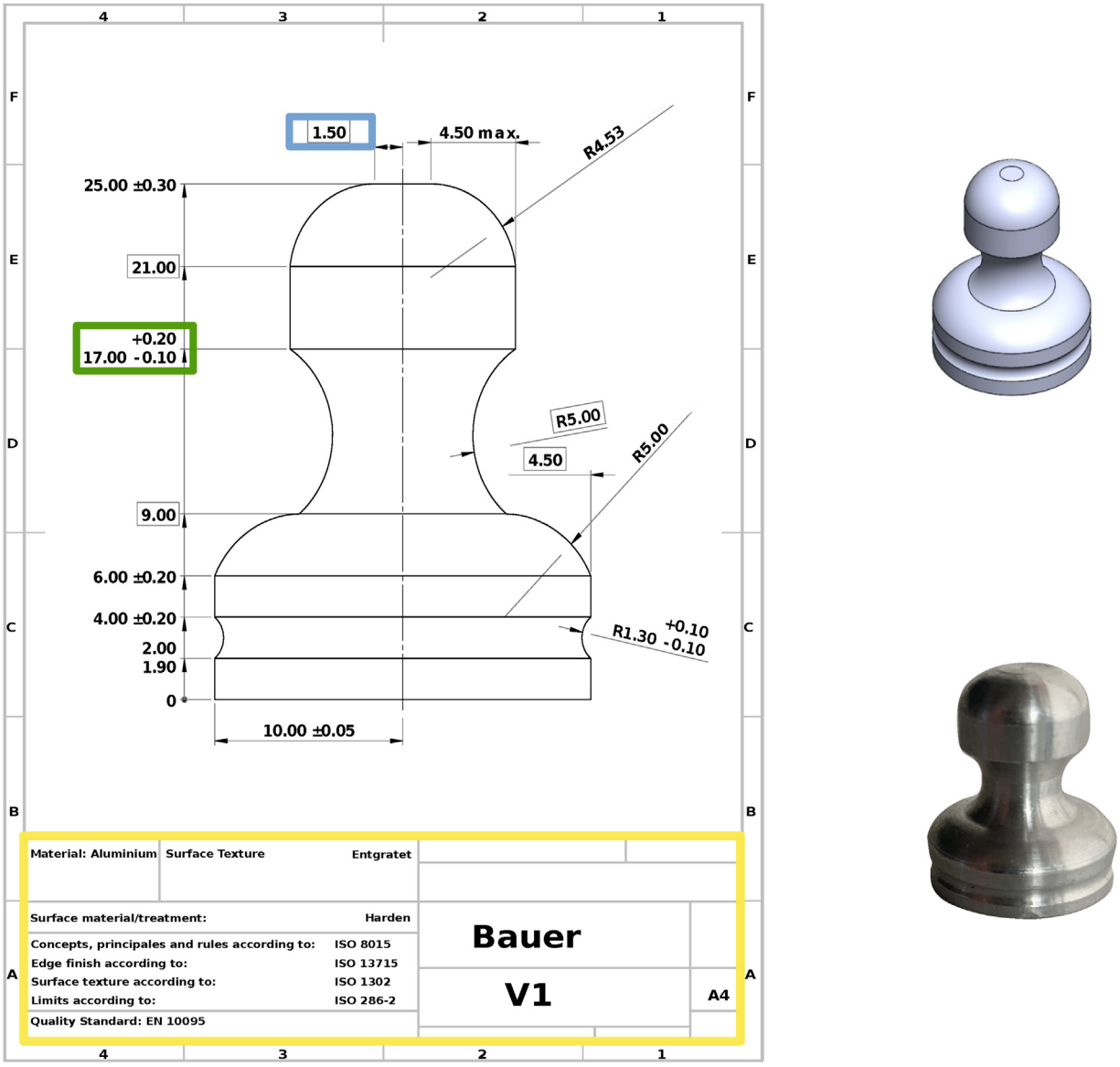

Fig. 1. Engineering drawing.

isfied, as well as default dimensioning requirements, if these are not explicitly stated in the ED. Integrating this information into a continuous (semi-)automated production process can, among others, facilitate automating quality control by the means of automated measurement. There are approaches to include all of the additional information regarding dimensioning and tolerances in the CAD model, e.g., product and manufacturing information. However, it is still common practice to include this kind of information exclusively in the EDs. The transformation to CAD models as well as the extraction of information has been a well researched topic for the last decades. However, there is still no ready-to-use approach to effectively address the problem of integrating dimensioning information from EDs into a production process.

Hence, the *DigiEDraw* approach described in this work aims at supporting quality control by providing an *end-to-end approach* for digitalization and integration of EDs. End-to-end means that this approach includes the upload of a drawing, the information extraction, as well as the actual integration in the process. It also refers to a solution that does not require training, is easy to use, and supports an employee, who – in the end – can always check for validity. A prototypical implementation, which is deployed in a real-life manufacturing scenario, is described in Section 4.

The focus of this work lies on mechanical EDs, in particular component drawings. Component drawings display a specific workpiece from different views as well as the specifications that should be applied, as opposed to other kinds of EDs, e.g., assembly drawings that show how different workpieces are combined. However, the principles of DigiEDraw should apply for other kinds of EDs as well.

Fig. 1 depicts the ED of a geometric object. The *dimensions* are usually noted directly at the corresponding *graphical element*. Auxiliary lines can be used to specify which structural element the specification refers to. The first value is called the *nominal dimension*. Its value should lie between the minimum and the maximum *tolerance*. This area is also called the *tolerance zone*. The area highlighted in green in Fig. 1 is a typical example of a dimension. The nominal value is 17.00. +0.20 is the maximum tolerance which leads to an upper deviation of 17.20. −0.10 is the minimal tolerance, which means that the lower deviation is 16.90 and the tolerance zone lies between 16.90 and 17.20. This combination of nominal values and tolerances is called a *dimension set*. The element high-

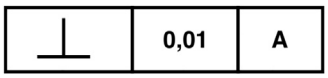

Fig. 2. Example of a geometrical dimension.

lighted in blue is a theoretically exact dimension, which means that this dimension has no tolerances. This is displayed with a border. If a dimension is not a theoretically exact dimension, and has no explicit tolerances, the standard tolerances apply. The standard tolerances are specified in the applicable regulatory documents and norms. In addition to dimensioning and size tolerances as shown in Fig. 1, *geometrical tolerances* describe the form or position of an element, as well as other *geometric features* such as *orientation* and *run out*. In Fig. 2, for example, a tolerance for perpendicularity is set. The whole dimension would be interpreted as follows: the perpendicularity of the specific element compared to part A cannot differ more than 0.01. In general, an ED can therefore include text, symbols as well as graphical elements.

The variability of representation is a challenge for automatic information extraction. A dimension can consist of only the nominal value, a complete dimension set with nominal value, maximum and minimum tolerance or a nominal value with only one of these tolerances. Geometrical dimensions can consist of different symbols and have to be set in relation to the feature they describe, i.e., which part of the workpiece they refer to. A dimension is only meaningful in relation to this additional information. The extraction process should therefore be designed to keep the elements in correct composition.

EDs can exist in multiple formats such as DXF, PDF, STEP, or image formats (TIFF, PNG). Existing approaches mostly use image formats as legacy EDs are usually available as scanned images only (Section 2). Pure CAD formats such as STEP often do not include additional information such as the tolerances. PDF includes the information needed for quality control and offer the benefit of providing graphical and textual elements in an already separated way. DigiEDraw focuses on digital PDF, as most traditional approaches work with either scanned or CAD format, but no approach has provided a solution for PDF based EDs so far, which should provide improved results as text recognition/OCR does not has to be applied. For the intended application of DigiEDraw only the textual information is needed, as we assume that a CAD model is present in addition to the ED. PDF extraction is a well researched topic (see Section 2). However, EDs differ from other PDF documents in that they include a combination of geometrical and textual elements, which do not follow a uniform structure and are spread over the drawing area. These elements can be horizontal as well as vertical or lie at an angle and can consist of one or multiple values.

DigiEDraw addresses these challenges based on the following research questions:

Research Question 1: *How can textual information, specifically dimensioning requirements, be automatically extracted from EDs, under the condition that each extracted dimension set consists of only one dimension, the respective tolerances, and additional information?*

The textual information can be extracted using available tools. However, this leads to single extracted elements, that are not part of dimension sets anymore. The main challenge is therefore to merge the separate values in order to get the associated dimension sets. As DigiEDraw operates on 2D EDs, this problem can be understood as a 2D clustering problem and tackled using standard clustering techniques. DigiEDraw opts for DBSCAN (Ester et al., 1996) as it does not rely on the number of clusters as input parameter.

Research Question 2: *How should the clustering parameters be set to achieve optimal clustering results, i.e. complete dimension sets and avoid over-clustering? Specifically*:

- **RQ 2.1:** *Which distance metric leads to optimal results?*
- **RQ 2.2:** *How should the DBSCAN parameters be set?*

To achieve optimal results, multiple parameters have to be fitted. In addition, we have to compute a distance metric that reflects the goal of merging elements back into their respective dimension sets.

The contributions of this paper are (i) employing clustering for merging logically connected elements that have been split while being extracted from a PDF document, including pre- and post-processing steps, (ii) an iterative approach to set the required parameters for clustering, (iii) the definition and calculation of a custom distance metric that takes the domain logic into account and therefore (iv) providing an end-to-end approach that can be used in the industry for quality control support.

Moreover, the paper provides a prototypical implementation including a user interface and production process models where DigiEDraw is integrated as support for quality assurance. This demonstrates how product and process quality can be increased based on the integration of EDs.

The remainder of the paper is structured as follows: Section 2 discusses existing approaches. The DigiEDraw approach is presented in Section 3. The prototypical implementation of DigiEDraw and the application in a manufacturing process are presented in Section 4. Section 5 provides a discussion of DigiEDraw results and Section 6 concludes this work.

2. Related work

Research on digitalizing EDs dates back to the late 1980s (e.g. Krause et al., 1989), but still remains a challenging task, for example, regarding quality control in production. This is based on the observation that quality checking based on EDs is still mostly conducted in a manual way. In general, EDs are available as image format (i.e., raster graphics such as TIFF, PNG, and JPEG), CAD formats including DXF, DWG, IGES, and STEP, as well as vector graphic format such as PDF and SVG. Existing approaches work on one of these three format groups. Also, as an overarching observation existing approaches aim at (i) digitalizing EDs, using mostly graphical elements, or (ii) at extracting specific information in order to optimize the retrieval of EDs, the design and production process, or the product management. As mentioned in Section 1, EDs consist of text, symbols as well as graphical elements. A solution that provides complete digitalization would have to include all of these elements.

Hence, existing approaches can be categorized by their input format as well as their focus, i.e., textual elements, graphical elements, symbols or a combination (cf. Table 1). Approaches focusing on textual elements can be further divided. *Meta Information* includes papers that extract specific information, e.g., the version number or other information found in the drawing tables. Category *Dimensions, Aspect* includes approaches that extract dimension information, i.e., dimensions and tolerances, but describe only one part, e.g., how to detect dimension boxes, where afterwards OCR can be applied or only the OCR itself. Category *End-to-End*, by contrast, refers to approaches that offer a solution starting from an ED until the integration of the textual information into the application. Note that also the category *Combination* contains end-to-end approaches. In the following, we discuss existing work along Table 1.

2.1. Scanned/image-based drawings:

The first approaches to digitize EDs focused on scanned images and therefore raster graphics. They consist of algorithms for text/graphic segmentation (Lu, 2002), symbol recognition

Table 1
Overview of related work.

	Textual elements			Symbols	Graphical elements	Combination
	Meta Information	Dimensions, Aspect	End-to-End			
Scanned	Ondrejcek et al. (2009) Banerjee et al. (2016) Lu et al. (2008) Das et al. (2018)	Lu (2002) Das and Langrana (1997) Lai and Kasturi (1994) Habed and Boufama (1999) Dori and Velkovitch (1998)		Archibald et al. (1995) Elyan et al. (2018) Elyan et al. (2020)	Vaxiviére and Tombre (1994) Ablameyko et al. (2002) Krause et al. (1989) Fonseca et al. (2005)	Mani et al. (2020) Kang et al. (2019) Rahul et al. (2019) Van Daele et al. (2019)
DXF/STEP/IGES	Jiang and Feng (2010) Zhang et al. (2012) Cao et al. (2005)				Sukimin and Haron (2008) Ye et al. (2009)	Prabhu (2002) Prabhu et al. (2001) Zhang and Li (2014)
PDF/SVG			DigiEDraw		Kasimov et al. (2015)	Hoang et al. (2016)

(Archibald et al., 1995), vectorization of raster graphics (Ablameyko et al., 2002), text recognition (Dori and Velkovitch, 1998), optical character recognition (OCR) (Brown et al., 1988; Lu, 1995), and combinations of these. A comprehensive overview regarding techniques for the digitalization of raster image EDs and the related use-cases is given by Henderson (2014), Moreno-García et al. (2018), and Tombre (1998). Nowadays work on this format continues, even though EDs are usually not stored in image format anymore. However, a large number of legacy EDs remains, which are mainly scanned. Recent approaches mainly build on neural networks, specifically convolutional neural networks or hybrid approaches which contain neural networks as well as traditional segmentation approaches (e.g., Elyan et al., 2018, 2020; Kang et al., 2019; Mani et al., 2020; Moreno-García et al., 2018; Rahul et al., 2019; Van Daele et al., 2019). In the following specific approaches are briefly explained, along the categories seen in Table 1.

2.1.1. Textual elements

In regards to *Meta Information*, Ondrejcek et al. (2009) aim at discovering links between scanned EDs and CAD models by analyzing ED features. Banerjee et al. (2016) search and analyze drawing names to find matching drawings and Lu et al. (2008) extract knowledge from tables. The tables are detected by analyzing the layout structure and matching it to a standard structure. This is then used to obtain meta information about a drawing. All of these approaches also use OCR in order to digitalize the letters and numbers. To decide which OCR engines should be used, algorithms to analyze if a drawing includes handwritten or machine written numbers can be used (Das et al., 2018). These approaches have in common that their focus on the drawing tables and search for very specific information.

Several approaches focus on *Dimensions*. Lu (2002) detect areas of dimension sets and separate them from the graphical elements, which is then used as input for OCR algorithms. Das and Langrana (1997), Lai and Kasturi (1994), and Habed and Boufama (1999) also look for dimension sets, using classic image analysis techniques (e.g., connected components), and analyzing the lines and arrows in the drawing. For recognition of the characters, existing OCR algorithms are used. The OCR step itself is not described, Das and Langrana (1997), for example, already assume that vectorization was performed before and OCR will lead to correct results. Dori and Velkovitch (1998) similarly detects dimension set areas, but it also includes a recognition algorithm. However, it is only able to detect 23 different characters, no symbols and only if the dimension set does not have a border.

2.1.2. Symbols

Archibald et al. (1995) present an approach to find and categorize symbols by matching these to templates. Nowadays, often deep learning and convolutional neural networks (CNN) are used for symbol detection (Elyan et al., 2018, 2020). These approaches focus on piping and instrumentation (P/ID) diagrams, a specific kind of engineering drawings, which relies heavily on symbol usage. However, symbol recognition in images is used in other domains such as music notes (Pacha et al., 2018).

2.1.3. Graphical elements

For a full digitalization of EDs, the recognition of graphical elements is necessary. This is mostly done using vectorization techniques. Mostly, these drawings are then transformed into CAD format (Vaxiviére and Tombre, 1994; Ablameyko et al., 2002; Krause et al., 1989). Fonseca et al. (2005) aim at facilitating the retrieval of EDs by analyzing the graphical elements, extracting features and the topology, and thereby optimizing the search for similar drawings.

2.1.4. Combination

These approaches aim at fully digitalizing EDs, including text, symbols as well as graphical elements. Mani et al. (2020), Kang et al. (2019), and Rahul et al. (2019) focus on a full digitalization of P/ID diagrams, where symbol detection is important, as well as text and connection recognition. All of these approaches are end-to-end approaches using a combination of traditional techniques as well as deep learning. Van Daele et al. (2019) support design processes by searching drawings, summarizing the most important features and finding similar ones. This is done using CNN as well as reasoning-based methods.

The main challenge for image-based EDs is the quality of the input files, as it can vary greatly and accordingly leads to different results. Furthermore, in order to use neural network approaches, which are the most promising, significant amounts of annotated EDs are needed as training data. In addition, training might require experts and is time-consuming.

2.2. CAD format (DXF, DWG, STEP, IGES)

Several approaches focus on developing 3D models from CAD files or extracting specific information. When dealing with CAD drawings, the information is already in machine readable, textual form, but has to be parsed and combined to obtain useful information. Therefore the description and categorization in Table 1 refers to what the elements are in the ED, i.e., text or graphical, and not their physical representation, as in this case, all elements are stored in textual form.

2.2.1. Textual elements

Jiang and Feng (2010) and Zhang et al. (2012) use DWG drawings to obtain product management data from the drawing tables in order to facilitate product management. Cao et al. (2005) similarly focus on product information to improve versioning management of EDs. Therefore all of these approaches belong to the *Meta Information* category.

As these formats are already in digital form, symbol detection does not have to be performed. Therefore no related work can be found in this category.

2.2.2. Graphical elements

Sukimin and Haron (2008) focus on extracting production relevant data such as the volume of a workpiece from the description of the graphical elements. Ye et al. (2009) describe a method to construct 3D models out of DXF drawings by simulating the human process of understanding EDs and combining different views of a workpiece.

2.2.3. Combination

Prabhu (2002) and Prabhu et al. (2001) extract graphical features, manufacturing information as well as information about the dimensions to improve the CAD/CAM linking. Zhang and Li (2014) aim at combining information from an ED with additional data stored in a database, i.e., information extraction, and visually display graphical elements. Therefore this approach also combines textual and graphical elements.

Even though the just described approaches, contain dimensioning information, CAD documents commonly do not contain additional information such as the tolerances and applicable regulatory guidelines, as these are often manually noted when the CAD models are transformed into EDs.

2.3. Vector drawings/PDF

Nowadays EDs are often stored as vector drawings, which can be processed digitally. Previous approaches focus on extracting either the graphical elements or facilitating the management of these EDs.

2.3.1. Graphical elements

Kasimov et al. (2015) describe a system for content-based retrieval of vector drawings based on a graph matching problem. This includes, for example, specifying a geometric feature, which is then used to search for drawings containing this feature.

2.3.2. Combination

Hoang et al. (2016) introduce an approach to automatically extract structural information as well as relationships from vector-based EDs. This also includes detecting symbols. Therefore this approach is categorized as *Combination*.

To the best of our knowledge there exists no approach to extract textual elements, specifically dimensioning information, from vector drawings or EDs in digital PDF format.

2.4. PDF information extraction

Multiple papers deal with the analysis and extraction of information from PDF documents in general (Bast and Korzen, 2017; Yuan et al., 2006; Adrian et al., 2017) or focus on specific documents, e.g., scientific papers (Bui et al., 2016; Wang et al., 2019) or health care related documents (Parizi et al., 2018; Li et al., 2019; Harmata et al., 2017). These approaches analyse pixels, words, spatial or logical connections. Ferres et al. (2018) also focus on extracting information not only from digital but also scanned PDF documents by including an OCR library. Similarly, Tomovic et al. (2020) use OCR techniques and even combine multiple OCR engines to generate robust input for classifying document segments. Corrêa and Zander (2017) give an overview of papers and tools regarding the extraction of tables from PDF documents. One of these tools is “texus” (Rastan et al., 2018) where the table is firstly detected by looking for the typical table structure and afterwards the content is extracted row by row, column by column. Another research focus are visually rich documents, i.e., documents where text and images are combined. Wei et al. (2020) propose an approach to extract specific information for example the address, from resumes as well as invoices that often contain images as well as text. “Layout-analysis” provides algorithms to detect and classify components of such documents, e.g., text paragraphs, headlines, figures, tables or algorithms (Prüša and Fujiyoshi, 2017; Zhong et al., 2019; Li et al., 2018). Hansen et al. (2019) focus on extracting all non-textual components, i.e., tables, algorithms, figures. Shi et al. (2019) take figures from medical papers and split these into the respective sub figures for better analysis. Morris et al. (2019) focus on the extraction of text from figures in PDF documents using OCR and neural networks.

Overall, a variety of approaches exists to extract different kinds of information from PDF documents. However, all of the covered document types differ from EDs as they mostly consist of only textual elements that are written in lines, or if containing figures and tables they follow a similar structure (e.g., resumes and invoices). Lastly, they do not contain elements that are overlapping, unequally distributed over the page and are in vertical or diagonal orientation. EDs, by contrast, are a combination of textual and graphical elements, which are distributed in the ED in no machine-understandable order.

2.5. Grouping and clustering

The specific challenge of DigiEDraw is to merge textual elements extracted from ED into complete dimension sets. The general challenge of grouping extracted elements has also been addressed by existing approaches. Habed and Boufama (1999) use the proximity and thickness of elements to find graphical elements and the associated text elements. Van Daele et al. (2019) employ clustering to combine graphical elements into logical clusters and to partition the ED into different views. The main difference to DigiEDraw is that it aims at deciding automatically whether textual elements that can be positioned in different distances and orientation belong to the same dimension set. This decision depends at least partly on domain knowledge which in turn has to be considered in a potential clustering approach.

2.6. Conclusion

Even though information extraction from EDs has been researched for more than three decades, many challenges remain. Approaches using machine learning such as neural networks are the only end-to-end approaches and seem promising, but also have weaknesses especially concerning the availability of annotated training data and effort of training (Moreno-García et al., 2018). The envisioned DigiEDraw approach, by contrast, does not require any training. Regarding vector drawings and digital PDFs, approaches focus mainly on graphical elements. However, none of the existing approaches take dimensions and tolerances into account. Previous works referring to PDF extraction focus on more structured and homogeneous document types, also containing mainly textual elements. In addition, source code is not available for most of the aforementioned papers. Therefore, we conclude that there is no end-to-end approach yet for automatically extracting textual elements, specifically dimension sets, from EDs.

3. DigiEDraw algorithms

The goal of DigiEDraw is to extract textual elements from EDs into complete and coherent dimensions sets. Consider Fig. 3, left side, where preprocessing yields elements 7 and , 3 as separated boxes, although the elements obviously belong together (cf. Fig. 3, middle). Likewise, the tolerances provided by elements $+0$, 1 and -0, 1 in Fig. 3 are required to complete the actual dimension set as depicted in Fig. 3, right side. The challenge is to automatically

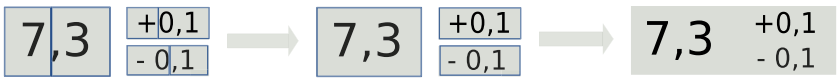

Fig. 3. Dimension set split up into three blocks and multiple words, clustered to achieve complete dimension set.

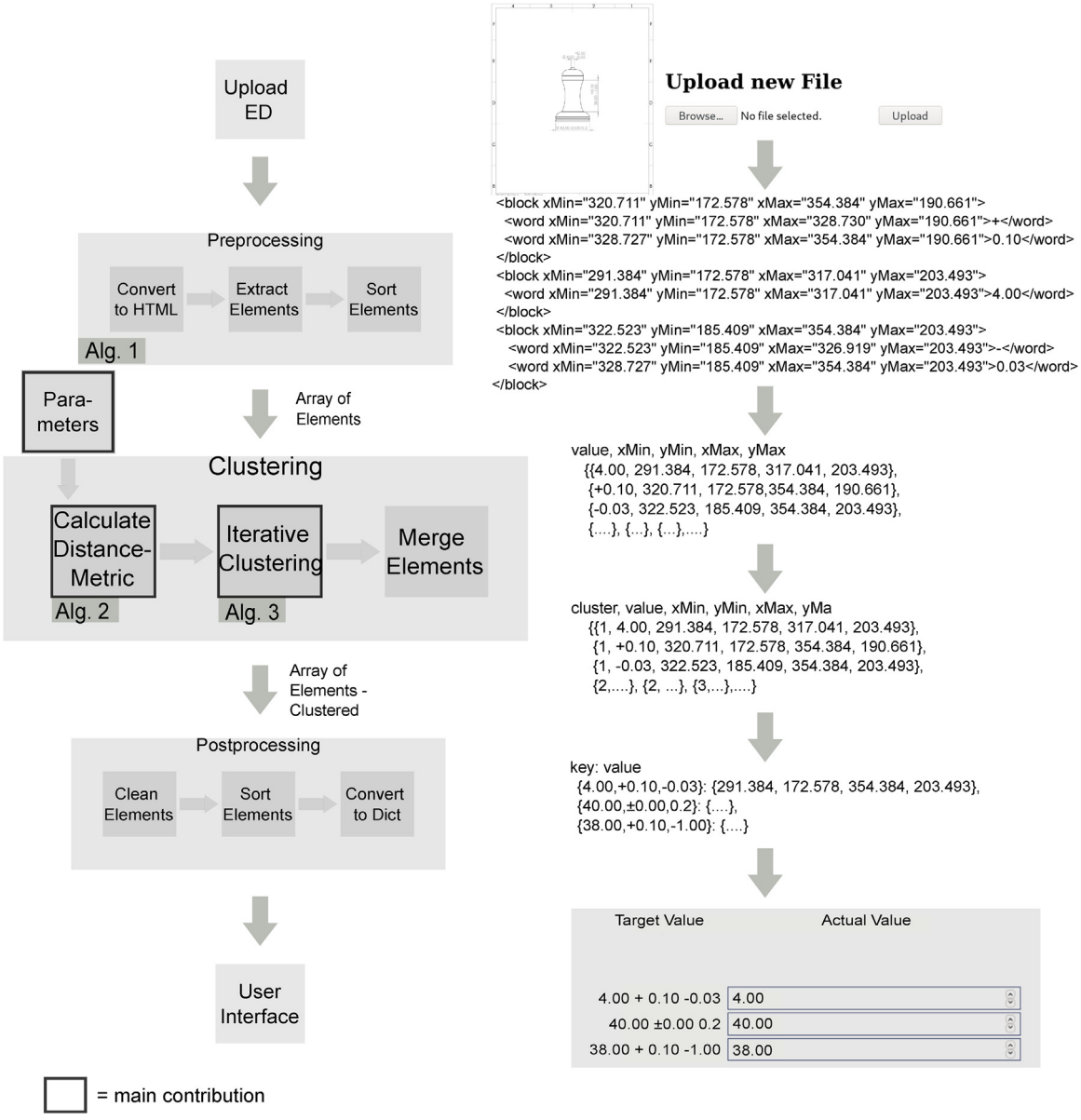

Fig. 4. Overview of the DigiEDraw algorithm and example.

determine and merge the elements that belong together in dimension sets. The idea of DigiEDraw is to employ clustering for this, i.e., to combine well-known clustering technique DBSCAN with a novel distance metric and an iterative parameter setting algorithm as core, embedded into specific pre- and postprocessing steps.

Fig. 4 provides an overview on the overall DigiEDraw approach (left side) as well as an illustrating example (right side). The input is the ED that is preprocessed at first (cf. Algorithm 1). The result is an array of elements that constitutes the input for the DigiEDraw core, i.e., the clustering of the elements into complete and coherent dimension sets (cf. Algorithm 3). Indicated by black borders in Fig. 4, the main conceptual DigiEDraw contributions comprise the iterative approach to set clustering parameters optimally in order to ensure a correct representation of dimensions as well as the definition of a novel distance metric (see Algorithm 2). The output here is an array of clustered elements which is then postprocessed, i.e., the resulting dimension sets are integrated into the manual quality control using a web service (cf. DigiEDraw prototype and production process models in Section 4).

3.1. Preprocessing

The preprocessing of the DigiEDraw approach starts by uploading an ED in PDF format that is converted into HTML. The reason is that HTML enables the extraction of all elements with their respective coordinates. There are different python libraries for reading

PDF available, e.g., PyPDF2,[1] textract,[2] and Tika.[3] . We obtained the best results using the `pdftotext` script which is part of the poppler-tools[4] integrated in unix systems. The resulting files after conversion to HTML consist of blocks and words. Each word consists of one or multiple characters or digits. For each element the coordinates of the bounding box are given as well. In some cases, one dimension set is extracted into exactly one HTML block. However, mostly, a dimension set is split into multiple blocks. Fig. 3, for example, shows a single dimension set that was split into three blocks (illustrated by the blue rectangles) by the conversion. Each of these blocks consists of multiple words, visualized by the dark blue borders in the example. For further processing, the textual elements, as well as the associated coordinates are read from the HTML file and stored in an array consisting of multiple subarrays, each representing one block, which also contain arrays, each representing one word. For the clustering process only the outer coordinates, i.e. the bounding box, are needed. Therefore the maximum and minimum x-coordinates and y-coordinates for each block are extracted and stored in the array. One element in the example depicted in Fig. 4 (right side) consists of textual value 4.00 together with the associated coordinates *xMin="291.384", yMin="172.578", xMax="317.041", yMax="203.493"*. The elements inside a block are not always in the correct order. To sort the elements, we determine if a block is horizontal or vertical by comparing the ratio of x to y values. If the block is horizontal, the words within a block are ordered by x-coordinates using the built-in sort function. The same procedure is done for vertical boxes, using y-coordinates. Algorithm 1 describes the necessary preprocessing steps.

Algorithm 1. DigiEDraw preprocessing.

Input: digital PDF ED
Output: array of bounding boxes
convert PDF to HTML
parse HTML file
find all blocks
for block in blocks **do**
  find all words
  for word in words **do**
get text element
get bounding box (min x, max x, min y, max y)
add text and bounding box to array *bounding_boxes* ▷ Array of array
  end for
end for
for element in *bounding_boxes* **do**
  check if element is vertical or horizontal
  if horizontal **then**
sort by x-coordinates
  end if
  if vertical **then**
sort by y-coordinates
  end if
end for
return sorted array *bounding boxes*

3.2. Clustering

The goal is to obtain complete dimension sets based on the information extracted from the input ED. Therefore, we have to determine which elements belong to the same dimension set, but have been extracted separately. As illustrated by the example in Fig. 3, not only numbers can be subject to merge (e.g., 7 and 3), but also associated tolerances (e.g., $+0$, 1 and -0, 1). Additionally, the elements can be positioned in a horizontal, vertical or diagonal way.

We tested several approaches to combine extracted elements into logically connected groups, without using training data. A "brute force" approach is to hard-code the relationships between elements in the ED, i.e., elements that are next to each other are combined. A more refined approach is to combine elements using regular expressions. However, both approaches did not yield accurate results as the structure of dimensions and tolerances can vary greatly. Clustering algorithms, by contrast, seem more promising with respect to merging logically connected elements, without requiring extensive training data. We tested k-means (Faber, 1994), using elbow plots to specify the amounts of clusters, OPTICS (Ankerst et al., 1999), and DBSCAN (Ester et al., 1996) where DBSCAN does not require to set the number of clusters as input. The results for k-means and OPTICS were less accurate than for DBSCAN. As a conclusion, we opted for clustering, and in particular, for DBSCAN to be employed by DigiEDraw.

Open questions are how to define the distance metric and how to set the parameters. DBSCAN implementations are available for most programming languages.[5]

For DBSCAN, the user sets parameter *MinPts*, which defines how many points should at least be in a cluster and *epsilon*, which defines the radius for each cluster, as well as the appropriate distance metric, which is used to calculate how "similar" or "close" elements are to each other. The algorithm then analyzes all points, grouping the ones together that are within the specified radius. For DigiEDraw, a new distance metric is defined that takes into account spatial proximity as well as domain knowledge. An additional challenge is the setting of *epsilon*, as this value has a great impact of the result, but has to be set dynamically, as it can be different for each ED. This issue was solved by using an iterative approach, increasing the value of *epsilon* until a threshold is reached – the stopping criterion – which is an indicator that the optimal result has been achieved. The envisioned result of the DigiEDraw clustering is an array of sorted elements, each element being assigned to a cluster. Consider the example in Fig. 4, right side. Three elements are assigned the same cluster number 1. This means that they form a dimension set and are stored as key value pair together with the coordinates. The coordinates of the dimension set are determined by the minimum of *xMin, yMin* and the maximum of *xMax, yMax* coordinates over all its elements.

The *distance metric* is used to calculate which boxes are close or similar to each other. Several metrics are available such as the Euclidean distance. However, as DigiEDraw works with bounding boxes instead of points, existing metrics are not sufficient. For humans it is easily recognizable which parts belong together. However, it is more complex to achieve this automatically.

The basic measure is the distance between the two nearest corners of two bounding boxes. To account for the boxes being close or just single edges near to each other, the distances of the two nearest edges of the boxes are averaged to obtain the distance. If the boxes intersect, this distance is set to zero, as these boxes will most likely belong to one dimension set. Conversely, if two boxes are parallel, the distance is increased as these boxes are likely not part of a dimension set and therefore should not be in one cluster.

Definition 1 *(Distance metric).* Let box_a and box_b be two bounding boxes of elements a and b extracted from an ED. The distance between a and b is defined as the average of the smallest and second smallest distance between a corner point of bounding box box_a and a corner point of bounding box box_b. If box_a and box_b intersect, which is checked using the separating axis theorem (Gottschalk et al., 1996), the distance is set to zero. If the elements are paral-

[1] https://github.com/mstamy2/PyPDF2
[2] https://github.com/deanmalmgren/textract
[3] https://tika.apache.org/
[4] https://poppler.freedesktop.org/

[5] The DigiEDraw prototype uses the `scikit-learn` clustering library (Pedregosa et al., 2011).

lel, which is determined by comparing orientation and alignment of box_a and box_b, the distance is increased. This is done for all bounding boxes of all elements extracted from the ED. The result is captured within a the distance matrix. The calculation is presented in pseudo code in Algorithm 2.

We check for intersection using the separating axis theorem (Gottschalk et al., 1996) which says that two boxes cannot overlap if there is an axis that separates them. In practice, this is done by taking two boxes and comparing the respective bottom left and top right corners to see if they overlap on either axis. To check for parallelism, first the orientation of the boxes is defined more precisely than in the preprocessing. As in the preprocessing, the ratio of length and width of a box can suggest the orientation of the box. In this case, we have three categories: vertical, horizontal or "not defined", which is characterized by similar length and width. The ratio which differentiates between these categories is determined empirically by looking at multiple sample EDs and measuring the ratio for vertical as well as horizontal boxes.

The orientation of the bounding boxes is calculated as follows:

$$horizontal = (xMax - xMin) > 1,3 * (yMax - yMin)$$

$$vertical = (yMax - yMin) > 1,3 * (xMax - xMin)$$

xMax is defined as the maximum x-coordinate, *xMin* accordingly refers to the minimum x-value of the respective bounding box. Correspondingly, *yMax* and *yMin* are the maximum and minimum value for the y-axis. The remaining boxes are set as "not defined". After the orientation is defined, the alignment of the two boxes in relation to each other is analyzed, i.e., are the boxes above each other or next to each other. If the boxes are above each other and horizontally aligned, then they are marked as being parallel. The same is done if the two boxes are next to each other and vertically aligned. The distance to the parallel box is then increased by 100. The exact value of the increase does not influence the results as long as it it exceeds the *epsilon* value. This is done for each box in relation to all other boxes, resulting in a distance matrix, which can be used in the `scikit`-DBSCAN implementation.

Algorithm 2. Calculate distance metric.

Input: array of bounding boxes
Output: distance matrix
for box_a in *bounding boxes* **do**
  calculate orientation
  for box_b in *bounding boxes* **do** ▷ Calculate distance from every box to other boxes
    calculate smallest distance between boxes a,b
    calculate 2nd smallest distance between boxes a,b
    check intersection using separating axis theorem
    if a,b intersect **then**
      set $distance_{a,b}$ to 0
    end if
    check parallelism by comparing orientation, alignment of boxes
    if a,b are parallel **then**
      increase $distance_{a,b}$ by 100
    end if
    $distance_{a,b} = (smallest\ \ distance + 2nd\ \ smallest\ \ distance)/2$ ▷ Calculate average distance
    store $distance_{a,b}$ in *distance matrix*
  end for
end for
make array $dist_{\min}$, including all distances, sorted in ascending order ▷ Needed as input for *epsilon*
return *distance matrix*, $dist_{\min}$ ▷ The distances from each box to the other boxes are stored in a matrix

The *MinPts* value relates to the minimum amount of points in the *epsilon* distance of a point to constitute a cluster. In this setting *MinPts* is set to 1, as it is possible that a dimension set consists of a single box.

Determining the optimal *epsilon* value is more complex. At first, we manually adjusted the value such that the first sample ED showed optimal results. As more EDs were analyzed, it became obvious that this value is not optimal for all EDs. Therefore, we opt for setting the value dynamically. Ester et al. (1996) and Sander et al. (1998) propose an interactive approach to set the *epsilon* value, i.e., computing a distance graph for all points and letting the user choose the threshold value. This approach did not work in this case, as the graph suggests higher *epsilon* values than needed, which leads to over clustering. Ozkok and Celik (2017) also use a k-nearest neighbor graph to determine the optimal setting. Schubert et al. (2017) note that the *epsilon* value is depending on the distance metric and the domain, but should generally be as small as possible.

In regard to EDs, the *epsilon* value depends on multiple variables and can differ for each ED. An iterative approach was taken to determine the optimal *epsilon* value. For all boxes the distances to all other boxes are calculated and stored in ascending order in the array $dist_{\min}$. The first *epsilon* value is the value of the minimal distance, i.e., the distance between the nearest boxes. In each iteration this value is increased to the next bigger distance value. For defining the stopping criterion, the clustering result was evaluated in regards to the following criteria:

- Davis-Bouldin index (Davies and Bouldin, 1979): The Davis-Bouldin index measures the average similarity measure of each cluster with its most similar cluster. Similarity is defined as the ratio of within-cluster distances to between-cluster distances. The lower to zero, the better the separation of clusters, as the clusters are farther apart.
- Calinski-Harabasz index (Caliński and Harabasz, 1974): The Calinski-Harabsz index is defined as the ratio of sum of between-cluster dispersion and inter-cluster dispersion. The higher the value, the better defined are the clusters.
- Silhouettes coefficient (Rousseeuw, 1987): Similar to Calinski-Harabasz index, the higher the value, the better defined are the clusters. It is calculated by comparing the intra-cluster distance and the mean nearest-cluster distance (where the element is not a part of), to achieve clusters that are cohesive and distinct.

These values are internal quality criteria of clustering algorithms and take into account how well clusters are defined and separated. These criteria were chosen because these values do not need the ground truth, which is not available at this point. The Calinski-Harabasz index as well as the silhouettes coefficient should get bigger if the higher *epsilon* leads to better separated clusters. Conversely, the Davis-Bouldin index should get closer to zero. In each run the current value of these indices is compared to the values in the previous run. If these values change in the opposite direction it can be assumed that the best result was already achieved. Therefore the iterations are stopped and the *epsilon* value of the previous run is set as the optimal value. This was tried with each of the three parameters as well as combinations of these. The silhouette coefficient leads to the best results and is therefore used as stopping criterion. The clustering process stops as soon as the stopping criterion evaluates to true or the highest distance value is reached.

Stopping criterion: Clustering is applied with increasing *epsilon* values until the stopping criterion is reached. As stopping criterion the *silhouettes coefficient* (Rousseeuw, 1987) is used, which is an internal quality value, referring to the cluster definition. The value of the silhouettes coefficient rises if the clusters are more defined. Therefore we continue the iterative clustering process, until the coefficient gets smaller than in the step before, which is a sign of the cluster definition getting worse. At this point, the loop is stopped and the end result will be obtained by running DBSCAN

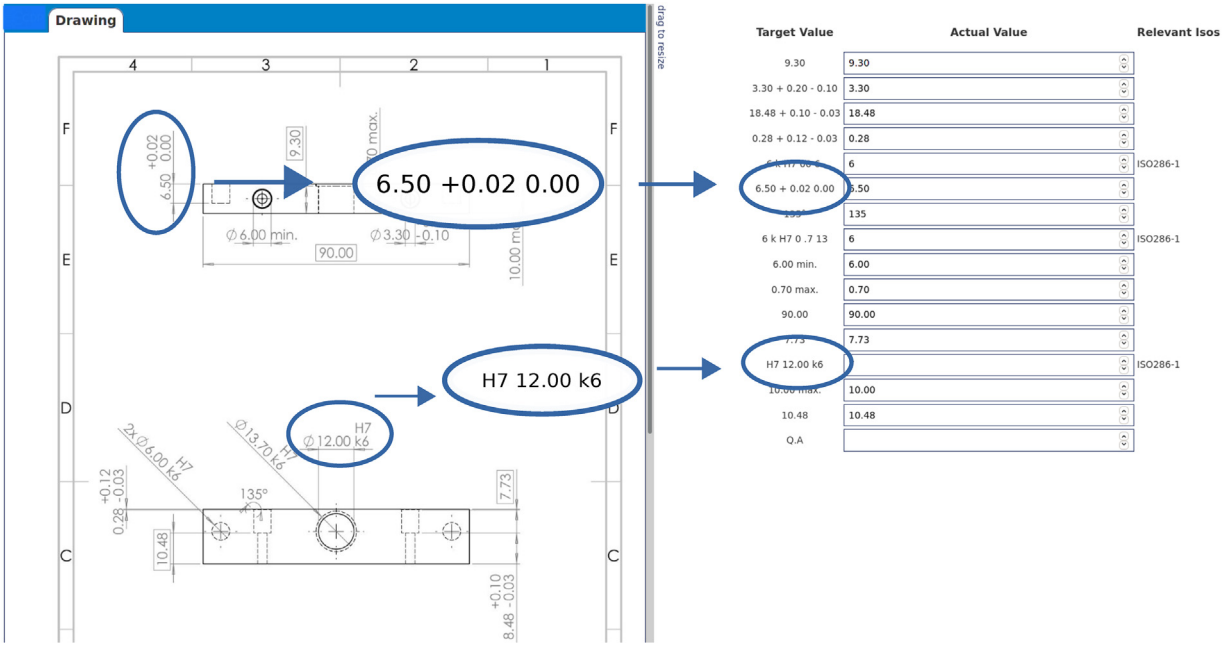

Fig. 5. User interface featuring ED 'Elevator Bottom'.

with *epsilon* being the penultimate value - the value where the silhouettes coefficient was highest.

Algorithm 3. DigiEDraw clustering.

Input: array of bounding boxes, distance matrix, dist_min
Output: dictionary of dimension sets
get array *bounding boxes*, *distance matrix*, $dist_{\min}$
run DBSCAN(*distance matrix*, *epsilon* = *smallest* $dist_{\min}$, *MinPts* = 1) ▷ First iteration of clustering
if $sh < sh_{old}$ ▷ Silhouettes coefficient is used as stopping criterion **then**
  stopping_criterion = true ▷ If Silhouettes coefficient gets smaller, stopping threshold is reached
end if
while not stopping_criterion ▷ Continue clustering while stopping criterion is not true
  run DBSCAN(*distance matrix*, *epsilon* = *next* $dist_{\min}$, *MinPts* = 1) ▷ *epsilon* = next value of $dist_{\min}$
  if $sh < sh_{old}$ **then**
stopping_criterion = true
  end if
  if epsilon = $nn_{\max}$ ▷ If *epsilon* reaches maximum distance, clustering is stopped **then**
end clustering
  end if
end while
run DBSCAN with penultimate *epsilon* ▷ The value before reaching the stopping criterion is used
return array of clusters ▷ Each cluster containing one to multiple elements

The clustering algorithm can be seen in Algorithm 3. We take the results of the preprocessing (Algorithm 1) and the calculation of the distance matrix (Algorithm 2) and run the DBSCAN implementation as long as the stopping criterion is not met. The result (array of elements, assigned to clusters) is then returned to continue with postprocessing. Overall, the clustering result depends on the quality of the ED, i.e., whether or not the ED was designed according to existing standards, e.g., with respect to distance between elements, the *epsilon* value, *MinPts*, and the employed distance metric.

3.3. Postprocessing

The array of elements with assigned cluster numbers (see Fig. 4, right side for the running example) resulting from Algorithm 3 is postprocessed in several steps. At first, data cleaning is performed, using regular expressions. In detail, the cleaning involves filtering for expressions that are not relevant for dimensions. This step can be adapted to fit particular needs and scenarios. In addition, the elements are sorted again. If the bounding boxes of the elements are determined as horizontal, sorting is performed based on the x-value and if vertical by the y-value, to ensure that after merging the clusters, the elements are in correct order. The last step converts the array of elements into a dictionary which can then be stored in a key value store.[6] Fig. 4, right side, shows the resulting key value set for the running example. Entry {4.00, +0.10, −0.03}: {320.711, 172.578, 354.384, 203.493}, for example, refers to a dimension set containing elements 4.00, +0.10, 0.03 with the associated coordinates of the overall bound box. The resulting dimension sets are then postprocessed using regular expressions, stored in a database, and on request by the user shown via the DigiEDraw user interface (cf. Fig. 5).

4. Evaluation

DigiEDraw is evaluated with respect to its feasibility, the recall and precision of its algorithms, and its application.

4.1. Feasibility – prototypical implementation

Algorithms 1–3 are implemented within the DigiEDraw prototype which consists of an extraction tool[7] and a webservice including the user interface.[8] The DigiEDraw user interface is depicted in Fig. 5. The ED of the workpiece is displayed on the left while the automatically extracted dimension sets are shown on the right hand side. Next to the dimensions, the user can input the manual measurements which are automatically stored in a database for further processing. The user can click on the input field next to the dimension, and the associated element is highlighted in the ED. Moreover, all references to associated regulatory standards and norms are extracted using regular expressions, noted on top of the

[6] DigiEDraw uses redis, https://redis.io/.
[7] https://github.com/DigiEDraw/extraction
[8] https://github.com/DigiEDraw/ui

UI and linked if available. Therefore, these standards can be easily accessed if necessary. The DigiEDraw user interface enables the user to get a clear view on all dimensions and directly input the associated measurements.

4.2. Validation – recall and precision

The quality of the clustering algorithm can be validated using external or internal validation. Internal validation was already discussed in Section 3, for the stopping criterion. External validation refers to the comparison to the ground truth, which in this case consists of the actual dimensions. This can only be validated manually, as the actual dimensions have to be extracted by hand. According to literature regarding external evaluation for density-based clustering methods, Ester et al. (1996) use visual inspection. Aliguliyev (2009) mention "accuracy" and "recall", which are used to calculate more complex evaluation measures. Ting (2010) define the measures "precision" and "recall", which are commonly used to evaluate information extraction systems. In this work these precision and recall measures are used.

Definition 2 *(Recall).* In the context of information retrieval, Ting (2010) defines recall as:

$$Recall\_[58] : = \frac{Total\ number\ of\ documents\ retrieved\ that\ are\ relevant}{Total\ number\ of\ relevant\ documents\ in\ the\ database}$$

$$Recall_{DE} := \frac{extracted\_relevant\_dimension\_sets}{all\_relevant\_dimension\_sets}$$

where extracted_relevant_dimension_sets denotes the number of relevant dimension sets extracted from the ED and all_relevant_dimension_sets denotes the number of all relevant dimension sets in the ED.

In order to calculate this metric, the actual correct dimension sets, as displayed in the ED, as well as the correctly extracted, i.e. complete, sets are counted manually.

Definition 3 *(Precision).* Ting (2010) define precision as:

$$Precision\_[58] : = \frac{Total\ number\ of\ documents\ retrieved\ that\ are\ relevant}{Total\ number\ of\ documents\ that\ are\ retrieved}$$

For this paper, the value is adapted to:

$$Precision_{DE} := \frac{extracted\_relevant\_dimension\_sets}{all\_extracted\_elements}$$

where extracted_relevant_dimension_sets denotes the number of relevant dimension sets extracted from the ED and all_extracted_elements denotes all elements extracted by DigiEDraw.

This specifies how much useful information could be extracted out of all the information that was retrieved.

In order to calculate precision, the correctly extracted sets are the same as for the recall calculation. The total number of extracted elements are counted automatically.

Fig. 6 shows cutouts of four example EDs, i.e., "Gripper", "Aufspannung", "Aufspannung_Ecke" and "Adapterplatte". These EDs in full, as well as "Elevator Bottom" and "Halter" can also be downloaded from the git repository. An additional ED has been obtained by a company partner and is not publicly available.

Table 2
Evaluation results.

Drawing name	$Recall_{DE}$	$Precision_{DE}$	Iterations	eps-value
Gripper	0.82	0.6	4	4.3
Elevator Bottom	0.93	0.88	4	4.3
Aufspannung	0.85	0.58	3	4.15
Adapterplatte	0.93	0.87	5	8.58
GV12	0.88	0.76	4	4.88
Halter	0.85	0.73	5	5.06
Aufspannung_Ecke	0.91	0.67	4	4.2

Table 2 shows the validation results based on the seven example EDs in terms of recall and precision as well as the number of iterations and the used epsilon value.

The EDs chosen for the evaluation (cf. Fig. 6) cover a variety of possible component EDs, as all main parts (dimensions, size tolerances, geometrical tolerances, tables, multiple graphical elements) as well as the main challenges (e.g. combination of graphical and textual elements, unequally spaced out over the ED space, textual elements in different orientations) are part of these EDs and they differ in terms of quantity of elements and distribution as well as arrangement of these. The average recall is 0.88 and the average precision is 0.73. The results are discussed in more detail in Section 5.

4.3. Application

Fig. 7 depicts the process models (modeled using Business Process Modeling and Notation (BPMN)[9]) for producing the workpiece shown in Fig. 1. More precisely, Fig. 7a depicts the production process for a single workpiece. The process with manual quality control includes subprocess *Version 1* (cf. Fig. 7b) and the production process with a facilitated and optimized quality control using DigiEDraw includes subprocess *Version 2* (cf. Fig. 7c). For both processes, the actual production part is the same. For the quality control in Fig. 7b, an employee manually measures the dimensions of the workpiece and checks whether they are within the tolerance range specified in the corresponding ED (cf. Fig. 1). This quality control can be facilitated and optimized through DigiEDraw, i.e., by two system-based tasks `Display Dimensions in UI` and `Compare Measurements` as depicted in Fig. 7c. In both scenarios, the employee has to have the ED available. For Fig. 7b, the ED is read manually, whereas in Fig. 7c the ED has to be uploaded to the DigiEDraw webservice before the production process starts. For using DigiEDraw no additional knowledge or training time is necessary. The user simply has to upload the ED and is then provided with the UI shown in Fig. 5. The user still has to decide manually which features should be measured, which also acts as a plausibility check for the extracted dimensions.

Currently, DigiEDraw facilitates and optimizes manual measuring in production processes. However, DigiEDraw could also be used as part of an automated measurement process, i.e., by providing the input for a measuring program. In addition, DigiEDraw can help in the design phase by providing feedback if the elements are too close together or overlap and are therefore not only harder to be extracted automatically, but also lead to less readable EDs for humans.

5. Results and discussion

In this section, we summarize the results along research questions **RQ 1** and **RQ 2** as set out in the introduction. Then we

[9] `bpmn.org`

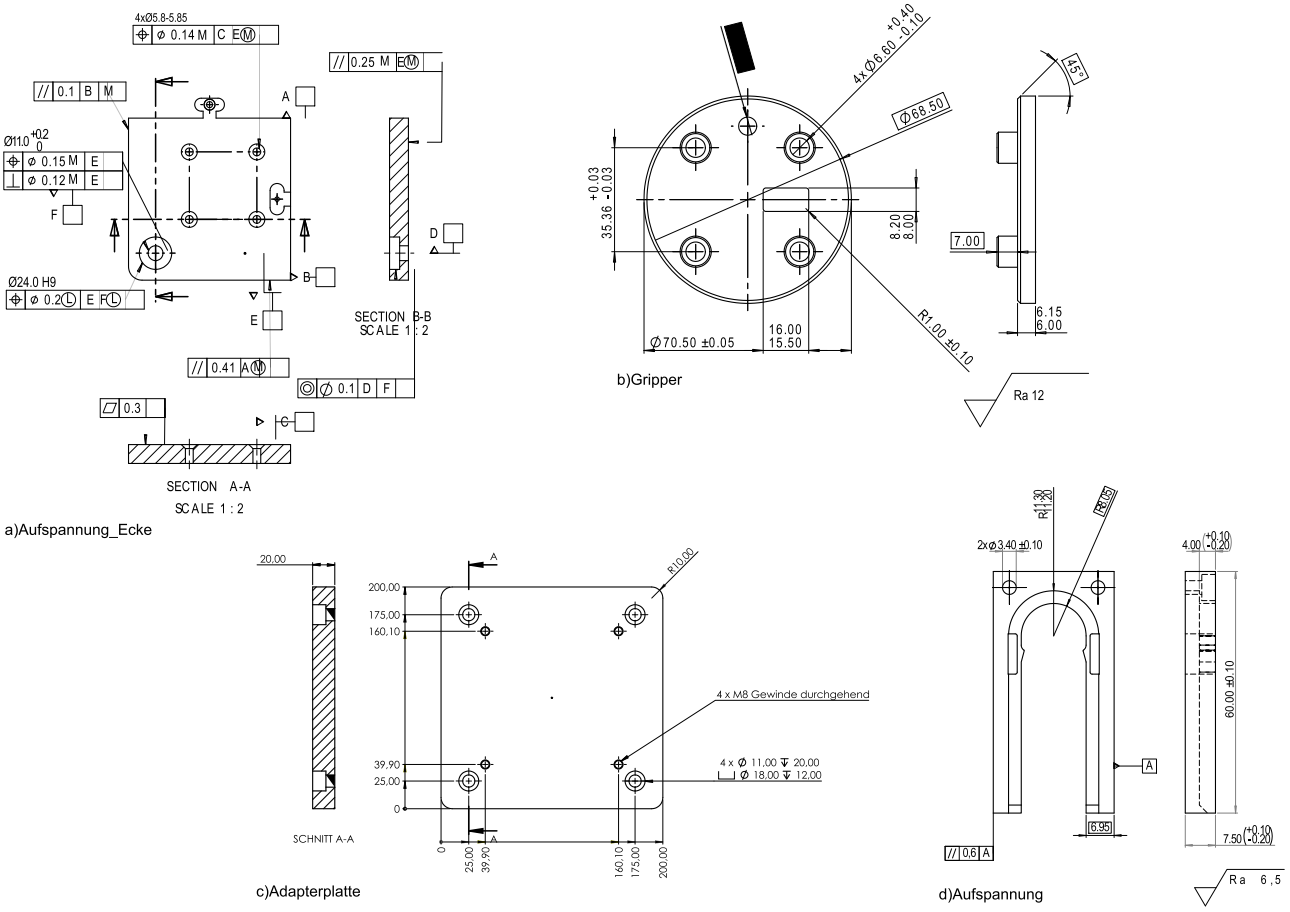

Fig. 6. Cutouts of three EDs.

formulate guidelines on EDs and discuss an extended application of DigiEDraw.

RQ 1: *How can textual information, specifically dimensioning requirements, be automatically extracted from EDs, under the condition that each extracted dimension set consists of only one dimension, the respective tolerances, and additional information?*

Existing approaches target different ED formats. This paper focuses on PDF format. Algorithm 1 realizes the preprocessing of EDs by extracting HTML elements, filtering and sorting the textual elements. After the elements are extracted, the dimension sets have to be recomposed. This is achieved through clustering using DBSCAN (cf. Algorithm 3) in an iterative way until the best possible results are achieved. Other solutions were also tried: converting the PDF into image format and using OCR to extract the dimension led to inferior results. Similarly, instead of clustering, regular expressions and a brute force approach provided worse results. Afterwards, postprocessing differentiates between dimension elements, text referring to regulatory standards and other textual elements and provides this additional information in form of a user interface to assist employees with manual quality control.

RQ 2 *How should the clustering parameters be set to achieve optimal clustering results, i.e. complete dimension sets and avoid over-clustering?*

- **RQ 2.1:** *Which distance metric leads to optimal results?*
- **RQ 2.2:** *How should the DBSCAN parameters be set?*

We propose a distance metric based on the average distance between the two nearest points of the bounding boxes, taking into account parallel and intersecting elements. Aside from the distance metric, the parameter influencing the result of DBSCAN the most, is the *epsilon* value. We propose an iterative approach, increasing the *epsilon* value in each run, before a stopping criterion is reached. As stopping criterion the silhouette coefficient is used, which measures the relation between the inner-cluster distance and the between-cluster distance. For evaluation purposes recall Recall_{DE} and precision Precision_{DE} are used. Table 2 shows the results of the evaluation. The recall values are between 0.82 and 0.93, whereas the precision values lie between 0.58 and 0.88. The average recall is 0.88, meaning that 88% of all relevant information has been extracted. The average precision is 0.73, meaning that of all extracted elements 73% are indeed dimension sets. The fluctuation in these values could be a result of different conditions such as the layout of the drawing and in particular if design guidelines regarding, e.g., minimal distance have been adhered to. The precision values depend not only on the clustering of the values, but also on the postprocessing where the elements are filtered. These values may seem partly low. For the described use case, a 100% precision is not essential, as there is always a user involved, who still has to measure the features manually. This means that the user interface might still show unrelated values, but these unrelated values have decreased significantly, achieving a better overview for the user. However, if DigiEDraw should be used to provide input for an automatic measuring program, without human involvement, higher precision values should be obtained. In future work, we will therefore focus on increasing recall and precision by learning from the user interaction. No substantial fluctuations were observed for the number of iterations and the eps-value. This can be explained by the fact that no extreme cases, where values are very far apart or close together, were part of the evaluation set. However, even though

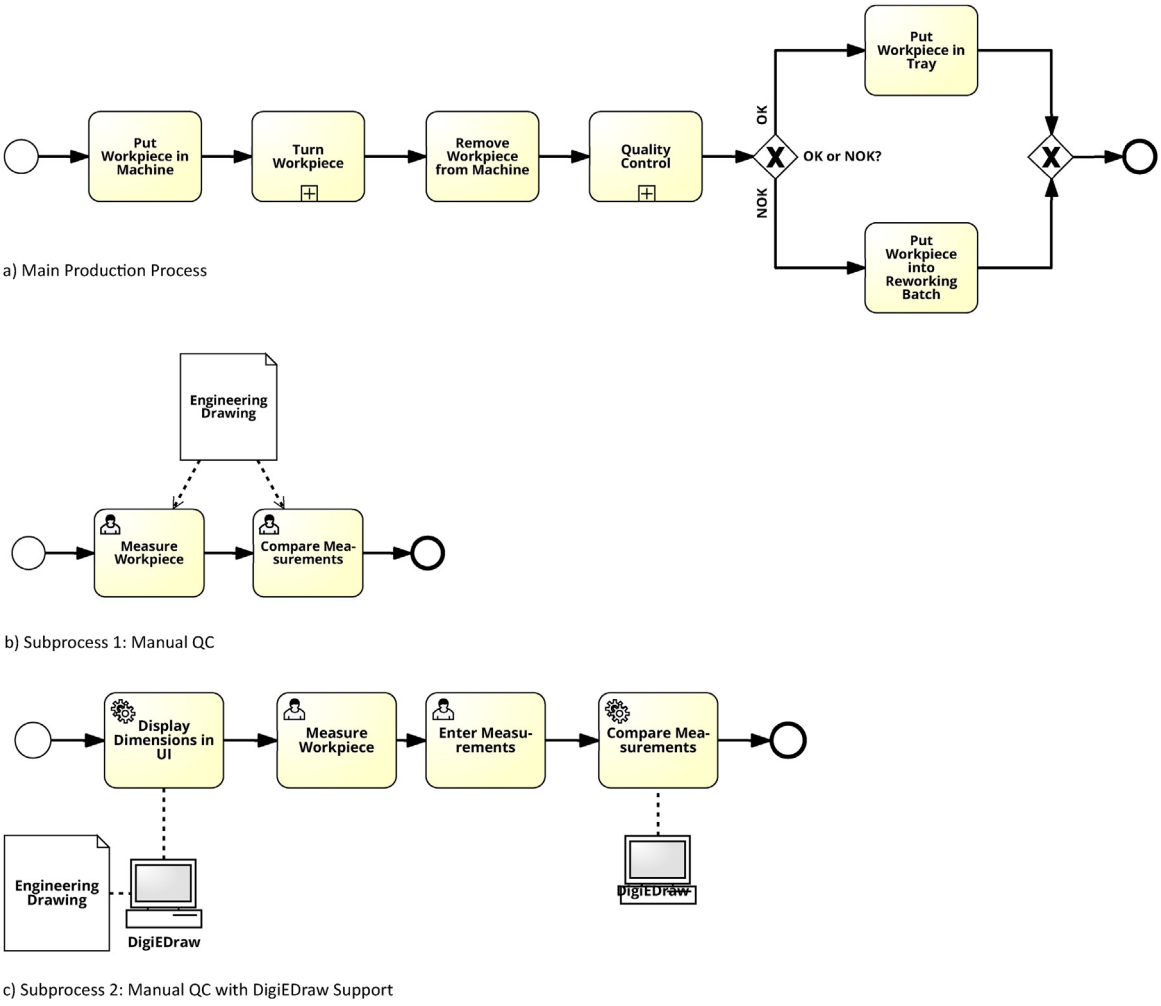

Fig. 7. Production process (modeled using Signavio©): (a) the main production process, (b) with manual quality control and (c) with facilitated and optimized quality control.

the range of variation does not seem large, even small changes can influence the recall and precision values.

Section 2 discusses approaches for scanned EDs as well as CAD format to digitize the graphical as well as textual elements. If the EDs are only available on paper, one of the mentioned approaches has to be used. However, these approaches may be problematic, as the quality heavily depends on the quality of the scan. In addition, most of these approaches only focus on one aspect and source code is mostly not available. Approaches for CAD formats are also limited, as it is not common to note dimensions in these models, but rather when transitioning the models to EDs. If PDF documents are available these issues can be avoided, and accurate results can be achieved by directly extracting from the PDF using DigiEDraw. DigiEDraw is providing an end-to-end approach, starting from uploading a drawing, to the integration of extracted values into a process. It is also easy to use and apply in industry, as it does not require programming skills or training time. Unique to DigiEDraw is also that only the pre- and postprocessing steps rely on domain-specific knowledge, as the clustering step itself relies on the distance metric. Thus, the DigiEDraw clustering approach can be adapted to other domains requiring only small adjustments.

5.1. Limitations

DigiEDraw is currently limited to EDs in digital PDF. However, DigiEDraw could be extended to include DXF as well as scanned EDs, by adding a DXF parser and an OCR library to the preprocessing stage. The approach is sensitive to noise, e.g., if dimensions overlap. Therefore the best results are achieved if the ED adheres to the standards and norms where, among others, spacing and font is specified. Currently, the approach was tested on component drawings, but can be adapted/extended to, for example, assembly drawings. Furthermore, the DigiEDraw approach is intended to be used in conjunction with manual quality control, therefore including another check of the extracted dimensions. The following structuring guidelines specify under which conditions the best results can be achieved.

5.2. ED structuring guidelines

The lessons learned from designing, implementing, validating, and applying DigiEDraw flow into the following guidelines. They state how EDs should be structured for achieving optimal results and supporting quality control in production processes:

- The ISO norms concerning the distances between elements are adhered to, in order to avoid overlapping.
- The ED is either in conventional portrait or landscape format.
- Diagonal text should be avoided, clear horizontal or vertical writing is optimal.
- Standard fonts are used.

If all of the guidelines are followed, DigiEDraw can even extract dimensions out of complex EDs. Additionally, well structured EDs are also easier to understand for the employees.

Section 4 discusses the application of DigiEDraw to facilitate and optimize manual quality control in the production process. In addition, DigiEDraw can be used as quality control for the ED itself. This is motivated as follows: extraction results may vary according to the compliance to standards. It is important that the different values are organized and specifically no values are overlapping or stacked upon each other. If this is the case, the extraction process cannot work properly. Thus, DigiEDraw can be used to assess the extractability of an ED, while it is still in the design phase. The design engineer can then immediately react and adapt the ED accordingly, facilitating reading for humans as well as for DigiEDraw. In addition, DigiEDraw can be used to generate an automatic measurement program, by extracting dimensioning information, only requiring the employee to check the extracted elements and edit these if necessary, similar to the approach mentioned by Rica et al. (2020). This would be a more efficient approach than manually extracting all information. A measurement program could then be used to fully automate the quality control process. Zeleny et al. (2017) also use a clustering algorithm for web page pre-processing using HTML boxes as input. This could constitute another application, in a completely different domain.

6. Conclusion

Today, EDs are mainly used as contractual basis as well as to provide additional information, in particular information about tolerances. However, there is still no approach to incorporate these EDs automatically in the production process. This paper proposes DigiEDraw to extract dimensions from EDs using DBSCAN. The dimensions can then be used for quality control tasks. The development of DigiEDraw shows that it is possible to extract dimensioning information from EDs with a recall of over 88% and a precision of 73%. DigiEDraw was applied for facilitating and optimizing manual quality control in a real-world production process. Additional application scenarios such as the automatic creation of measurement programs are conceivable. To the best of our knowledge, this is the first end-to-end approach where dimensions are extracted from a PDF ED.

DigiEDraw is currently limited to EDs in digital PDF. However, DigiEDraw could be extended to also include DXF as well as scanned EDs, by adding a DXF parser and an OCR library to the preprocessing stage. The best results are achieved if the ED adheres to the guidelines e.g., standards are complied with. Currently, the approach was tested on component drawings, but can be adapted/extended to, for example, assembly drawings.

In future work, more parameters influencing the recall and precision of DigiEDraw will be explored. This could be achieved by learning from the user interaction, i.e. allowing the user to mark unrelated values, storing these values and therefore learning for future drawings. In addition, the relationship between font size and clustering parameters will be examined and included into the algorithm. This could lead to a more domain specific approach. At all, it seems promising to include more semantics such as specific positions or areas and their meaning, potentially in combination with object recognition. In addition, we will explore how it can be distinguished between essential and non-essential information, e.g., is there a pattern which dimensions are essential for overall quality of a workpiece. Finally, we will explore additional application scenarios and types of EDs.

Conflict of interest

The authors declare that there is no conflict of interest.

Declaration of Competing Interest

The authors report no declarations of interest.

Acknowledgments

This work has been partially supported and funded by the Austrian Research Promotion Agency (FFG) via the “Austrian Competence Center for Digital Production” (CDP) under the contract number 854187.

References

Ablameyko, S., Bereishik, V., Frantskevich, O., Homenko, M., Paramonova, N., 2002]. A system for automatic recognition of engineering drawing entities. In: Proceedings. Fourteenth International Conference on Pattern Recognition, IEEE Comput. Soc, pp. 1157–1159, http://dx.doi.org/10.1109/ICPR.1998.711901 http://ieeexplore.ieee.org/document/711901/.

Adrian, W.T., Leone, N., Manna, M., Marte, C., 2017]. Document layout analysis for semantic information extraction. In: Esposito, F., Basili, R., Ferilli, S., Lisi, F.A. (Eds.), AI*IA 2017 Advances in Artificial Intelligence. Springer International Publishing, Cham, pp. 269–281, http://dx.doi.org/10.1007/978-3-319-70169-1_20.

Aliguliyev, R.M., 2009]. Performance evaluation of density-based clustering methods. Inf. Sci. 179, 3583–3602, http://dx.doi.org/10.1016/j.ins.2009.06.012.

Ankerst, M., Breunig, M.M., Kriegel, H.P., 1999]. OPTICS: ordering points to identify the clustering structure. In: Proceedings ACM SIGMOD International Conference on Management of Data, June 1–3, 1999, Philadelphia, Pennsylvania, USA, p. 12, http://dx.doi.org/10.1145/304182.304187.

Archibald, C., Kwok, P., Gros, C., 1995]. Automatic understanding of technical drawings: symbol extraction by geometric and morphological methods. Res. Comput. Robot Vis., 347–366, http://dx.doi.org/10.1142/9789812812483_0020.

Banerjee, P., Choudhary, S., Das, S., Majumdar, H., Roy, R., Chaudhuri, B.B., 2016]. Automatic hyperlinking of engineering drawing documents. Proceedings – 12th IAPR International Workshop on Document Analysis Systems, DAS 2016, 102–107, http://dx.doi.org/10.1109/DAS.2016.76.

Bast, H., Korzen, C., 2017]. A benchmark and evaluation for text extraction from PDF. 2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL), 1–10, http://dx.doi.org/10.1109/JCDL.2017.7991564.

Brown, R.M., Fay, T.H., Walker, C.L., 1988]. Handprinted symbol recognition system. Pattern Recognit. 21, 91–118, http://dx.doi.org/10.1016/0031-3203(88)90017-9.

Bui, D.D.A., Del Fiol, G., Jonnalagadda, S., 2016]. PDF text classification to leverage information extraction from publication reports. J. Biomed. Inform. 61, 141–148, http://dx.doi.org/10.1016/j.jbi.2016.03.026.

Caliński, T., Harabasz, J., 1974]. A dendrite method for cluster analysis. Commun. Stat. 3, 1–27, http://dx.doi.org/10.1080/03610927408827101.

Cao, Y., Li, H., Liang, Y., 2005]. Using engineering drawing interpretation for automatic detection of version information in CADD engineering drawing. Autom. Constr. 14, 361–367, http://dx.doi.org/10.1016/j.autcon.2004.08.004.

Corrêa, A.S., Zander, P.O., 2017]. Unleashing tabular content to open data: a survey on PDF table extraction methods and tools. In: Proceedings of the 18th Annual International Conference on Digital Government Research, ACM, Staten Island, NY, USA, pp. 54–63, http://dx.doi.org/10.1145/3085228.3085278.

Das, A.K., Langrana, N.A., 1997]. Recognition of dimension sets and integration with vectorized engineering drawings. Comput. Vis. Image Underst. 68, 90–108, http://dx.doi.org/10.1006/cviu.1997.0537.

Das, S., Banerjee, P., Seraogi, B., Majumder, H., Mukkamala, S., Roy, R., Chaudhuri, B.B., 2018]. Hand-written and machine-printed text classification in architecture, engineering and construction documents. Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR 2018, 546–551, http://dx.doi.org/10.1109/ICFHR-2018.2018.00101, ISBN: 9781538658758.

Davies, D.L., Bouldin, D.W., 1979]. A cluster separation measure. IEEE Trans. Pattern Anal. Mach. Intell. PAMI-1, 224–227, http://dx.doi.org/10.1109/TPAMI.1979.4766909.

Dori, D., Velkovitch, Y., 1998]. Segmentation and recognition of dimensioning text from engineering drawings. Comput. Vis. Image Underst. 69, 196–201, http://dx.doi.org/10.1006/cviu.1997.0585.

Elyan, E., Garcia, C.M., Jayne, C., 2018]. Symbols classification in engineering drawings. 2018 International Joint Conference on Neural Networks (IJCNN), 1–8, http://dx.doi.org/10.1109/IJCNN.2018.8489087, ISSN: 2161-4407.

Elyan, E., Jamieson, L., Ali-Gombe, A., 2020]. Deep learning for symbols detection and classification in engineering drawings. Neural Netw. 129, 91–102, http://dx.doi.org/10.1016/j.neunet.2020.05.025.

Ester, M., Kriegel, H.P., Sander, J., Xu, X., 1996]. A density-based algorithm for discovering clusters in large spatial databases with noise. In: Proceedings of the Second

International Conference on Knowledge Discovery and Data Mining, AAAI Press, Portland, Oregon, pp. 226–231.
Faber, V., 1994]. Clustering and the Continuous K-Means Algorithm, vol. 22. Los Alamos Science.
Ferres, D., Saggion, H., Ronzano, F., 2018]. PDFdigest: an adaptable layout-aware PDF-to-XML textual content extractor for scientific articles. LREC 2018, 6.
Fonseca, M.J., Ferreira, A., Jorge, J.A., 2005]. Content-based retrieval of technical drawings. Int. J. Comput. Appl. Technol. 23, 86, http://dx.doi.org/10.1504/ijcat.2005.006467.
Gottschalk, S., Lin, M.C., Manocha, D., 1996]. OBBTree: a hierarchical structure for rapid interference detection. In: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques – SIGGRAPH '96, ACM Press, pp. 171–180, http://dx.doi.org/10.1145/237170.237244.
Habed, A., Boufama, B., 1999]. Dimension sets detection in technical drawings. Vision Interface, 217–223.
Hansen, M., Pomp, A., Erki, K., Meisen, T., 2019]. Data-driven recognition and extraction of PDF document elements. Technologies 7, 65, http://dx.doi.org/10.3390/technologies7030065.
Harmata, S., Hofer-Schmitz, K., Nguyen, P.H., Quix, C., Bakiu, B., 2017]. Layout-aware semi-automatic information extraction for pharmaceutical documents. In: Da Silveira, M., Pruski, C., Schneider, R. (Eds.), Data Integration in the Life Sciences. Springer International Publishing, Cham, pp. 71–85, http://dx.doi.org/10.1007/978-3-319-69751-2_8.
Henderson, T.C., 2014]. Analysis of Engineering Drawings and Raster Map Images. Springer, New York, http://dx.doi.org/10.1007/978-1-4419-8167-7.
Hoang, X.L., Arroyo, E., Fay, A., 2016]. Automatische Analyse und Erkennung graphischer Inhalte von SVG-basierten Engineering-Dokumenten. At-Automatisierungstechnik 64, 133–146, http://dx.doi.org/10.1515/auto-2015-0089.
Jiang, Z., Feng, X., 2010]. An information extraction of title panel in engineering drawings and automatic generation system of three statistical tables A. The cell description of BOM information table. The data configuration of BOM information. 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), IEEE, vol. 1, http://dx.doi.org/10.1109/ICACTE.2010.5579014, pp. V1-297–V1-301.
Kang, S.O., Lee, E.B., Baek, H.K., 2019]. A digitization and conversion tool for imaged drawings to intelligent piping and instrumentation diagrams (P&ID). Energies 12, 2593, http://dx.doi.org/10.3390/en12132593.
Kasimov, D.R., Kuchuganov, A.V., Kuchuganov, V.N., 2015]. Individual strategies in the tasks of graphical retrieval of technical drawings. J. Vis. Lang. Comput. 28, 134–146, http://dx.doi.org/10.1016/j.jvlc.2014.12.010.
Krause, F.L., Jansen, H., Großmann, G., Spur, G., 1989]. Automatic scanning and interpretation of engineering drawings for CAD-processes. CIRP Ann. – Manuf. Technol. 38, 437–441, http://dx.doi.org/10.1016/S0007-8506(07)62741-3.
Labisch, S., Weber, C., 2008]. Technisches Zeichnen Selbstständig lernen und effektiv üben, 3rd ed. Viewegs Fachbücher der Technik, http://dx.doi.org/10.1007/978-3-8348-9273-7.
Lai, C.P., Kasturi, R., 1994]. Detection of dimension sets in engineering drawings. IEEE Trans. Pattern Anal. Mach. Intell. 16, 848–855, http://dx.doi.org/10.1109/34.308483.
Li, P., Jiang, X., Shatkay, H., 2019]. Figure and caption extraction from biomedical documents. Bioinformatics 35, 4381–4388, http://dx.doi.org/10.1093/bioinformatics/btz228.
Li, X.H., Yin, F., Liu, C.L., 2018]. Page object detection from PDF document images by deep structured prediction and supervised clustering. In: 2018 24th International Conference on Pattern Recognition (ICPR), IEEE, Beijing, pp. 3627–3632, http://dx.doi.org/10.1109/ICPR.2018.8546073.
Lu, T., Yang, Y., Yang, R., Cai, S., 2008]. Knowledge extraction from structured engineering drawings. Proceedings – 5th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2008, 415–419, http://dx.doi.org/10.1109/FSKD.2008.184, ISBN: 9780769533056.
Lu, Y., 1995]. Machine printed character segmentation – an overview. Pattern Recognit. 28, 67–80, http://dx.doi.org/10.1016/0031-3203(94)00068-W.
Lu, Z., 2002]. Detection of text regions from digital engineering drawings. IEEE Trans. Pattern Anal. Mach. Intell. 20, 431–439, http://dx.doi.org/10.1109/34.677283.
Mani, S., Haddad, M.A., Constantini, D., Douhard, W., Li, Q., Poirier, L., 2020]. Automatic digitization of engineering diagrams using deep learning and graph search. In: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), IEEE, Seattle, WA, USA, pp. 673–679, http://dx.doi.org/10.1109/CVPRW50498.2020.00096.
Moreno-García, C.F., Elyan, E., Jayne, C., 2018]. New trends on digitisation of complex engineering drawings. Neural Comput. Appl. 1, 1–18, http://dx.doi.org/10.1007/s00521-018-3583-1, ISBN: 0052101835.
Morris, D., Tang, P., Ewerth, R., 2019]. A neural approach for text extraction from scholarly figures. 2019 International Conference on Document Analysis and Recognition (ICDAR), 1438–1443, http://dx.doi.org/10.1109/ICDAR.2019.00231, ISSN: 2379-2140.
Ondrejcek, M., Kastner, J., Kooper, R., Bajcsy, P., 2009]. Information extraction from scanned engineering drawings. Aperture, 1–29.
Ozkok, F.O., Celik, M., 2017]. A new approach to determine EPS parameter of DBSCAN algorithm. Int. J. Intell. Syst. Appl. Eng. 5, 247–251, http://dx.doi.org/10.18201/ijisae.2017533899.
Pacha, A., Choi, K., Coüasnon, B., Ricquebourg, Y., Zanibbi, R., Eidenberger, H., 2018]. Handwritten music object detection: open issues and baseline results. 2018 13th IAPR International Workshop on Document Analysis Systems (DAS), 163–168, http://dx.doi.org/10.1109/DAS.2018.51.
Parizi, R.M., Guo, L., Bian, Y., Azmoodeh, A., Dehghantanha, A., Choo, K.K.R., 2018]. CyberPDF: smart and secure coordinate-based automated health PDF data batch extraction. 2018 IEEE/ACM International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), 106–111, http://dx.doi.org/10.1145/3278576.3281274.
Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., 2011. Scikit-learn: machine learning in Python. J. Mach. Learn. Res. 12, 2825–2830.
Prüša, D., Fujiyoshi, A., 2017]. Rank-reducing two-dimensional grammars for document layout analysis. 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), 1120–1125, http://dx.doi.org/10.1109/ICDAR.2017.185, ISSN: 2379-2140.
Prabhu, B.S., 2002]. Automatic extraction of manufacturable features from CADD models using syntactic pattern recognition techniques. Int. J. Prod. Res. 37, 1259–1281, http://dx.doi.org/10.1080/00207540191247.
Prabhu, B.S., Biswas, S., Pande, S.S., 2001]. Intelligent system for extraction of product data from CADD models. Comput. Ind. 44, 79–95, http://dx.doi.org/10.1016/S0166-3615(00)00073-7.
Rahul, R., Paliwal, S., Sharma, M., Vig, L., 2019]. Automatic Information Extraction from Piping and Instrumentation Diagrams, CoRR abs/1901.1. https://arxiv.org/pdf/1901.11383.pdf.
Rastan, R., Paik, H.Y., Shepherd, J., Ryu, S.H., Beheshti, A., 2018]. TEXUS: table extraction system for PDF documents. In: Wang, J., Cong, G., Chen, J., Qi, J. (Eds.), Databases Theory and Applications. Springer International Publishing, Cham, pp. 345–349, http://dx.doi.org/10.1007/978-3-319-92013-9_30.
Rica, E., Moreno-García, C.F., Álvarez, S., Serratosa, F., 2020]. Reducing human effort in engineering drawing validation. Comput. Ind. 117, 103198, http://dx.doi.org/10.1016/j.compind.2020.103198.
Rousseeuw, P.J., 1987]. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math. 20, 53–65, http://dx.doi.org/10.1016/0377-0427(87)90125-7.
Sander, J.R., Ester, M., Kriegel, H.P., Xu, X., 1998]. Density-based clustering in spatial databases: the algorithm GDBSCAN and its applications. Data Min. Knowl. Discov. 2, 169–194, http://dx.doi.org/10.1023/A:1009745219419.
Schubert, E., Sander, J., Ester, M., Kriegel, H.P., Xu, X., 2017]. DBSCAN revisited. ACM Trans. Database Syst. 42, 1–21, http://dx.doi.org/10.1145/3068335.
Shi, X., Wu, Y., Cao, H., Burns, G., Natarajan, P., 2019]. Layout-aware subfigure decomposition for complex figures in the biomedical literature. ICASSP 2019 – 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 1343–1347, http://dx.doi.org/10.1109/ICASSP.2019.8683824, ISSN: 2379-190X.
Sukimin, Z., Haron, H., 2008]. Geometric entities information for feature extraction of solid model based on DXF file. Proceedings – International Symposium on Information Technology 2008. ITSim, 1–5, http://dx.doi.org/10.1109/ITSIM.2008.4632024.
Ting, K.M., 2010]. Precision and recall. In: Sammut, C., Webb, G.I. (Eds.), Encyclopedia of Machine Learning. Springer US, Boston, MA, p. 781, http://dx.doi.org/10.1007/978-0-387-30164-8_652.
Tombre, K., 1998]. Analysis of engineering drawings: state of the art and challenges. In: Tombre, K., Chhabra, A.K. (Eds.), Graphics Recognition Algorithms and Systems. Springer, Berlin, Heidelberg, pp. 257–264, http://dx.doi.org/10.1007/3-540-64381-8_54.
Tomovic, S., Pavlovic, K., Bajceta, M., 2020]. Aligning document layouts extracted with different OCR engines with clustering approach. Egypt. Inform. J., http://dx.doi.org/10.1016/j.eij.2020.12.004.
Van Daele, D., Decleyre, N., Dubois, H., Meert, W., 2019]. An Automated Engineering Assistant: Learning Parsers for Technical Drawings, arXiv:1909.08552 [cs]. http://arxiv.org/abs/1909.08552.
Vaxiviére, P., Tombre, K., 1994. Knowledge organization and interpretation process in engineering drawing interpretation. Proc. IAPR Workshop on Document Analysis, 313–321.
Wang, Z., Beyette, D., Lin, J., Liu, J., 2019]. Extraction of math expressions from PDF documents based on unsupervised modeling of fonts. 2019 International Conference on Document Analysis and Recognition (ICDAR), 381–386, http://dx.doi.org/10.1109/ICDAR.2019.00068, ISSN: 2379-2140.
Wei, M., He, Y., Zhang, Q., 2020]. Robust layout-aware IE for visually rich documents with pre-trained language models. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, Virtual Event China, pp. 2367–2376, http://dx.doi.org/10.1145/3397271.3401442.
Ye, B., Liu, J., Wu, B., Wu, C., 2009]. New method of feature recognition from engineering drawings based on multi-granularity information acquisition. 6th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2009, vol. 5, 129–133, http://dx.doi.org/10.1109/FSKD.2009.802, ISBN: 9780769537351.
Yuan, F., Liu, B., Yu, G., 2006]. A study on information extraction from PDF files. In: Yeung, D.S., Liu, Z.Q., Wang, X.Z., Yan, H. (Eds.), Advances in Machine Learning and Cybernetics. Springer, Berlin, Heidelberg, pp. 258–267, http://dx.doi.org/10.1007/11739685_27.
Zeleny, J., Burget, R., Zendulka, J., 2017]. Box clustering segmentation: a new method for vision-based web page preprocessing. Inf. Process. Manag. 53, 735–750, http://dx.doi.org/10.1016/j.ipm.2017.02.002.
Zhang, H., Li, X., 2014]. Data extraction from DXF file and visual display. In: Stephanidis, C. (Ed.), HCI International 2014 – Posters' Extended Abstracts: International Conference, HCI International 2014, Proceedings, Part I. Heraklion, Crete, Greece,

June 22–27, 2014, pp. 286–291, http://dx.doi.org/10.1007/978-3-319-07857-1_51.

Zhang, J., Zhao, L., Hao, Y., 2012]. Multi-level block information extraction in engineering drawings based on depth-first algorithm. Adv. Mater. Res. 468–471, 2100–2103, http://dx.doi.org/10.4028/www.scientific.net/AMR.468-471.2100, ISSN: 10226680.

Zhong, X., Tang, J., Yepes, A.J., 2019]. PubLayNet: largest dataset ever for document layout analysis. 2019 International Conference on Document Analysis and Recognition (ICDAR), 1015–1022, http://dx.doi.org/10.1109/ICDAR.2019.00166, ISSN: 2379-2140.