

# Extracting decision models from digitally drawn or hand-drawn DMN images

Aurélie Leribaux  
 KU Leuven  
[aurelie-leribaux@hotmail.com](mailto:aurelie-leribaux@hotmail.com)

Caroline Heijmans  
 KU Leuven  
[caroline.heijmans@hotmail.be](mailto:caroline.heijmans@hotmail.be)

Alexandre Goossens  
 KU Leuven  
[alexandre.goossens@kuleuven.be](mailto:alexandre.goossens@kuleuven.be)

Jan Vanthienen  
 KU Leuven  
[jan.vanthienen@kuleuven.be](mailto:jan.vanthienen@kuleuven.be)

## Abstract

*Decision Model and Notation (DMN) models are used to model and automate operational decisions. Frequently, these DMN models are distributed as images within documents, either as screenshots or as pictures of hand-drawn models. This distribution method can result in the loss of the original source format. Re-using these images then entails the manual process of remodelling or redrawing them, a task that is both time-consuming and complex. In this study, deep learning techniques are employed to extract DMN models from both digitally drawn and hand-drawn DMN images. A substantial dataset was collected and annotated to train and test the diverse range of models. Subsequently, the work's outcome has been integrated into a DMN Computer Vision Tool application which can be used to reconstruct DMN source files based on hand-drawn sketches and digital images.*

**Keywords:** DMN, Computer Vision, Deep learning

## 1. Introduction

Operational decisions are executed on a daily basis and in high volumes making them important for companies to manage correctly (Vanthienen, 2021). With the Decision Model and Notation (DMN) standard (OMG, 2015), it is possible to model, manage, execute and automate operational decisions (Post et al., 2020). Given the benefits of using DMN models, multiple approaches have been proposed to reduce the time-consuming modeling task such as extraction methods from textual descriptions (Etikala et al., 2020; Quishpi et al., 2021), from process logs (Bazhenova et al., 2016; De Smedt et al., 2019), and from process models (Batoulis et al., 2015; Bazhenova et al., 2019).

In the initial conceptualization phase, DMN models are often hand-drawn with first sketches and still need to be manually modeled into a modeling software tool. Moreover, once modeled, the process and decision models are frequently distributed as images within documents rather than in their original source format (Antinori et al., 2022). It is also possible that the decision models are remodelled in other formats to allow easy distribution such as in presentation formats. This implies that for people who want to modify a DMN model without a source file, substantial time needs to be dedicated to remodel it in a modeling tool given that currently no tool exists to automatically extract DMN source models from images.

This study aims to address this issue by proposing a novel approach to reconstruct a source DMN file from both digitally drawn DMN images, these can be screenshots directly from the DMN modelling tool or other digital formats where a DMN model was drawn, and hand-drawn DMN images produced during brainstorming sessions or first sketches. A substantial dataset of both digitally drawn and hand-drawn DMN models was collected to train and evaluate a diverse range of deep learning models, including Region-based Convolutional Neural Networks (R-CNNs) (Girshick et al., 2014), and Optical Character Recognition (OCR) methodologies (Singh et al., 2012). Next, these models have been integrated into a comprehensive pipeline and tool for extracting decision logic from DMN images and converting them into machine-readable DMN files. Moreover, the dataset, models and the DMN vision tool are made available to the community.

The remainder of this paper is structured as follows: Section 2 deals with the related work followed by Section 3 introducing the problem statement. Section 4 elaborates on the methodology and Section 5 explains

the experiments and the results. Next, Section 6 discusses the results, limitations and future work. Finally, Section 7 concludes this paper.

## 2. Related Work

### 2.1. Extracting DMN models

DMN models can be used to model, communicate, execute and automate operational decisions. However, DMN models need to be modeled or extracted from something containing decisions such as text, process models or process logs.

**2.1.1. Text** A first approach to extract decision dependencies from textual descriptions was proposed by Etikala et al. (2020) with the use of pattern-based NLP. Using the same dataset, a later study also used pattern-based NLP to extract decision logic and FEEL from textual descriptions (Quishpi et al., 2021). The issue with pattern-based NLP, however, is that it does not scale well with unseen patterns in comparison to deep learning NLP models. A first approach to extract DMN models from textual descriptions using deep learning NLP models was proposed by Goossens, De Smedt, and Vanthienen (2023b) with the use of BERT models. Later on, the usage of GPT-3 models to extract decision dependencies and decision logic was investigated in (Goossens, De Smedt, and Vanthienen, 2023a) and (Goossens, Vandeveld, et al., 2023) respectively. A more in-depth discussion on the extraction of DMN models from texts was performed by Vanthienen and Goossens (2024).

**2.1.2. Process Models** According to the separation of concerns, processes and decisions should be modeled separately (Biard et al. (2015)). But sometimes this principle is not adhered to and process models contain decision logic. Batoulis et al. (2015) investigated how decision logic can be extracted from BPMN models whilst Bazhenova et al. (2019), proposed an approach to extract DMN models from process models.

**2.1.3. Process Logs** It is also possible to extract decision models from event logs to check how the decisions were executed in an organization just as can be done with process mining. This was first proposed with a decision mining algorithm in PROM in 2006 (Rozinat and van der Aalst, 2006). Later, Bazhenova et al. (2016) proposed an approach to extract decision models from process logs but it did not link the discovered decision models to the process models. This was addressed by De Smedt et al. (2019) where the discovered DMN

models are in line with the executed process and by Goossens et al. (2024) for object-centric event logs.

### 2.2. Image Analysis for BPMN and UML

Currently, there is no approach to extract DMN models from images, being from digitally drawn DMN images or hand-drawn DMN models, but there are approaches to extract BPMN models from images.

Pienikazek (2014) investigated how to extract business logic from BPMN models stored in a PDF format whilst Gantayat et al. (2018) proposed a deep learning approach to reconstruct BPMN models from images.

Antinori et al. (2022) introduce the open-source BPMNredrawer tool allowing to export BPMN images to BPMN source files but it does not support hand-drawn BPMN images. It makes use of the Detectron2 framework (Wu et al., 2019) to identify the shapes in the models and the Tesseract OCR tool (Smith, 2007) to extract the textual descriptions. Schäfer et al. (2021) investigated how sketches of BPMN models could be converted to BPMN source files and was extended with sketch2process which also supports the identification of textual labels, further improved the object and edge detection models and extended the dataset with more BPMN sketches (Schäfer et al., 2022) Lastly, the extraction of UML models from images was also studied by Karasneh and Chaudron (2013) and makes use of the traditional object and edge detection approaches paired with label detection.

## 3. Problem Statement, Research Goals & Assumptions

As said previously, the Decision Model and Notation (DMN) standard (OMG, 2015) allows to model, communicate, execute and automate operational decisions. A DMN model consists of a decision requirements diagram (DRD) which visualizes how a decision is structured with its required inputs and decisions whilst the decision logic is most commonly modeled with the use of decision tables which are read with if-then rules as they easily ensure correctness, completeness and consistency (Vanthienen et al., 1998). Despite recent advancements in extracting DMN models from various data sources such as original process models, Section 2 highlights a noticeable gap in the direct extraction of DMN models from images. Given that process-related information is often shared in image formats instead of the original source format (Antinori et al., 2022), a direct parallel can be drawn where decision-related information is also often shared in image formats rather than the original DMN XML

format. This presents a significant challenge as manually converting these initial images into DMN files is a time-consuming process, and hinders the efficiency of decision-making processes and organizational agility. Therefore, the research goals are:

1. Given a digitally-drawn or a hand-drawn DMN image:
  - (a) Extract a DRD from an image.
  - (b) Extract decision tables from an image.
  - (c) Align a DRD and decision tables together
  - (d) Integrate these components into a coherent and legible DMN file.
2. Develop the previous research outcomes into a user-friendly tool.

As this research is the first to explore DMN extraction from images and serves as a proof of concept rather than a fully developed solution, several initial simplifying assumptions are made:

- Only images showing either a single decision table or DRD are accepted as input, but not both simultaneously. This means a textual document containing text, decision tables and DRDs cannot be provided as input yet in this proof of concept and has to be split up.
- Sometimes, certain DMN elements are hidden within context menus of a modelling tool, e.g. hit policy details. This of course can not be analyzed with a screenshot of the tool. Therefore, context menus are not considered in this first research and only elements visually present in the figure are considered to be discoverable.
- Currently, only the most common visual elements of DRDs and decision tables are supported (i.e. decisions, dependencies, data sources, information items and decision rules). This means that newer extensions of the standards such as collections or annotations are currently not supported but this will be addressed in the future.
- For this first iteration a dataset of 62 models was manually annotated to investigate the feasibility of the approach. Even though there are more models available in a SAP-SAM dataset published by Sola et al. (2022), the main effort lies in the annotation of the models which is a time-consuming activity. In later iterations, only correct models from the SAP-SAM dataset identified by Corea et al. (2023) will be collected and annotated.
- Sometimes DMN models are accompanied with glossaries describing the domain knowledge for the DMN model. For this first iteration,

incorporating domain specific knowledge or glossaries is considered out of scope.

## 4. Methodology

Figure 1 visualizes the overall approach of this research. In the following subsection each step is explained in more detail:

1. **Dataset curation:** Both a digitally drawn and a hand-drawn dataset were collected and annotated by aggregating images from various studies and OMG (2015). These compiled datasets serve as the foundation for model construction and subsequent analyses.
2. **Image classification:** The images are manually classified into digitally or hand-drawn images. Furthermore, the images are also manually classified into visualizing a DRD or a decision table. It has been decided to not automate this step as this is straightforward and would require too much data to train a classification model.
3. **Image Preprocessing:** Following the image classification, this step involves the preprocessing the input images to enhance their quality and suitability for subsequent analysis.
4. **Decision requirements extraction:** This step focuses on extracting essential elements and dependencies from the decision requirements level depicted within the DRD images. For this purpose, the deep-learning framework Detectron2 (Wu et al., 2019) is used to train models capable of detecting objects within a DRD. Additionally, the Tesseract OCR framework (Smith, 2007) is utilized for text detection.
5. **Decision logic extraction:** Similarly to extracting DRDs, this phase aims at extracting the decision logic by also using Detectron2 (Wu et al., 2019) and Tesseract OCR (Smith, 2007) to train different models.
6. **Matching and DMN model generation:** Both the extracted DRD and extracted decision tables are matched together into a comprehensive DMN model in an XML format.
7. **Evaluation:** Lastly, the models and the DMN extraction pipeline are evaluated to ensure usability.

### 4.1. Dataset Curation

**4.1.1. Dataset Collection** A dataset has been collected from the official OMG DMN standard (OMG, 2015) and other academic papers. Table 1 presents an overview of both the digital and the hand-drawn

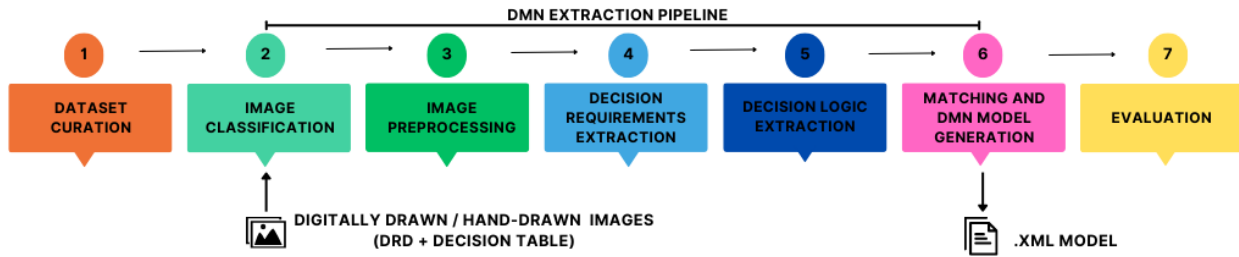


Figure 1. General Approach to extract DMN models from images

images. For the digital images, the DMN models were modeled in the Camunda tool <sup>1</sup> and screenshots were taken of both the DRDs and decision tables. Regarding the hand-drawn images, 9 different people were asked to draw on a piece of paper the collected digital DMN. This explains why the hand-drawn images dataset is larger than the digitally drawn dataset and also account for variations in styles.

Table 1. Dataset Overview

	Digital	Hand-drawn
<b>DRDs</b>	62	130
Decision	220	451
Input Data	228	479
Business Knowledge	58	118
Knowledge Source	39	75
Requirements	573	1109
Information	461	907
Knowledge	61	114
Authority	51	88
<b>Decision Table</b>	74	126
Decision/ Header	74	85
Rule	423	612
Hit Policy	74	115
Input	180	323
Output	83	131
Input entry	1242	2239
Output Entry	468	765

**4.1.2. Data Annotation** For all DRD and decision table images, each element mentioned in Table 1 was manually annotated in a Common Objects in Context (COCO) format (“COCO: Common Objects in Context”, n.d.) using the DataTorch platform (dataTorch, 2024) as is required for the Detectron2 framework (Wu et al., 2019).

Figure 2 shows how each element of a DRD is annotated. As can be seen, a blue box indicates an input

element, an orange box indicates a decision, a green box equals a knowledge source and a pink box a business knowledge. A similar approach was used to annotate the decision tables.

The requirements connecting the elements are annotated through keypoint annotations within each arrow. These keypoint annotations correspond to the initiation and termination points of each arrow. Additionally, two intermediary keypoints were introduced to account for directional variations in the digitally drawn DMN dataset, whereas four extra keypoints were used within the hand-drawn dataset.

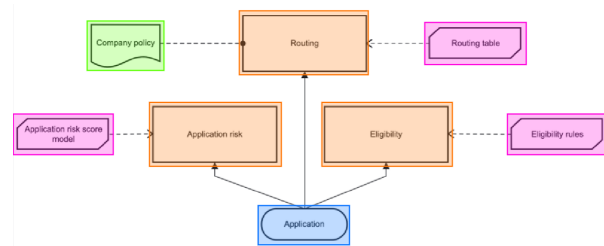


Figure 2. Annotated DRD for the elements

## 4.2. Image Classification & Preprocessing

**Image Classification:** As said previously, the images are manually classified into digital or hand-drawn images and subsequently manually classified into DRD and decision table. In the future, it would be possible to build an automatic classifier where maybe a pattern-based approach might be the easiest to implement given that DRDs and decision tables have such a great visual difference.

**Image Preprocessing:** In order for the models to more easily identify the relevant parts of an image, images need to be further preprocessed. This step is especially important to extract the textual annotations with OCR for both the DRDs and the decision tables.

Both the digital and hand-drawn images undergo a transparency enhancement processing step. This is to better distinguish the different elements and borders. The hand-drawn images are further preprocessed with a

<sup>1</sup><https://camunda.com>

grayscale conversion step (converting an image to gray) and thresholding step (forcing a pixel to either be white or black dependent on the gray value). Note that the digital images already are black and white and thus do not need these additional preprocessing steps.

### 4.3. Extracting DRDs

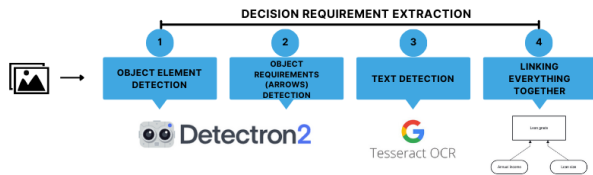


Figure 3. DRD Extraction Approach

Figure 3 visualizes the general approach to extract DRDs from images. The process starts by identifying on the image the different DRD elements, the requirements or arrows connecting them, and the associated text. For these 3 steps, object detection, keypoint detection and text detection models were trained. The Detectron2 framework (Wu et al., 2019), developed by Facebook AI Research, was used for both the object detection and the keypoint detection for the proof of concept given its good results on extracting BPMN models from images (Antinori et al., 2022). Pre-trained models from this framework were fine-tuned on the previously explained COCO datasets to ensure a good prediction performance to the task's requirements, despite the limited data. Additionally, state-of-the-art hyperparameters were employed for optimization of the task. The Text detection step uses the Tesseract OCR model (Smith, 2007).

The objective of the first object detection model is to accurately locate and classify the different elements within the DRD, namely the input data, decisions, business knowledge, and knowledge source elements. The model's output includes coordinates for each detected element along with its predicted label. This task was trained separately on both the digitally drawn DMN and hand-drawn datasets, resulting in two distinct models.

**4.3.1. Object Detection** For both types of images, a Faster Region-Convolutional Neural Network (F R-CNN) (ShaoqingRen, 2015) model was employed for this task. This model, designed for object detection tasks, consists of 50 layers in the Residual Network (ResNet) backbone (Kaiming et al., 2015). Additionally, it employs a Feature Pyramid Network (FPN) (Tsung-Yi et al., 2016) to efficiently extract hierarchical features, enabling the detection of objects of varying scales and sizes in images.

During training, images undergo further data augmentation such as resizing edges and horizontally flipping images with a certain probability, facilitated by the default data loader in Detectron2 (Wu et al., 2019). This loader prepares data in a format readily consumed by the model during training or evaluation. It has been decided to use the pre-trained model with default hyperparameter values given the limited dataset to perform an extensive hyperparameter optimization.

**4.3.2. Keypoint Detection** A DRD can contain information, knowledge, and authority requirements, depicted as arrows on a DRD. The detection model aims to precisely locate the origin and destination of these arrows by making keypoint predictions. For this task and both types of images, a Keypoint R-CNN model is trained and implemented using the Detectron2 framework (Wu et al., 2019). This model is specifically designed for keypoints recognition tasks and contains similar configurations as the Faster R-CNN model discussed in the previous subsection.

**4.3.3. Text Detection** For digital images, the textual labels are extracted from images using both Tesseract OCR (Smith, 2007) and OpenCV (OpenCV, 2023). OpenCV was mainly employed to further preprocess the textual labels so that Tesseract OCR could better extract the textual labels. For the hand-drawn images, a pretrained LSTM model for text recognition was employed.

### 4.4. Extracting Decision Tables

Regarding the extraction of decision tables, the same approach as in Figure 3 is used without the arrow detection steps.

**4.4.1. Object Detection** Given the inherent difficulty in accurately predicting table elements through object detection methodologies (Sachin Raja, 2022), two models with different setups were examined for both the digital and hand-drawn images:

1. A Faster R-CNN model (ShaoqingRen, 2015) that utilizes a ResNet-50 backbone network (Kaiming et al., 2015) combined with a Feature Pyramid Network (FPN) architecture (Tsung-Yi et al., 2016).
2. Mask R-CNN (matterport, 2018) might be preferred over Faster R-CNN for table structure detection tasks due to its ability to provide detailed segmentation information (Sachin Raja, 2022). A Mask R-CNN model architecture is

specifically designed for instance segmentation tasks, containing both object detection and pixel-wise segmentation (matterport, 2018). Therefore, potentially predicting table cells and structures more accurately. The model investigated combines a ResNet consisted of 50 layers as backbone network (Kaiming et al., 2015), with a FPN (Tsung-Yi et al., 2016) to extract hierarchical features from input images.

During training, the images undergo the same data augmentation as for DRDs.

**4.4.2. Text Detection** Using the detected table structure, the next step concerns the recognition of text which uses the same methodology as for text detection with DRDs.

#### 4.5. Constructing a complete DMN model

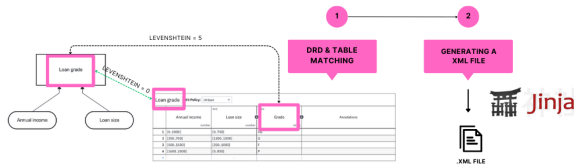


Figure 4. Merging into a complete DMN Model

Figure 4 shows the respective steps to be followed for the generation of a DMN model in a XML format. First, the DRD and decision table(s) are linked. To ensure reliable linking, table headers or, if absent, the output labels are compared the decision’s labels. To ensure that the correct tables are linked to the correct decisions, Levenshtein distances between all pairs are calculated and the minimum is retained.

Finally, the DMN model information is represented in a XML file, using a Jinja template. This template serves as a structured blueprint in accordance with the prescribed standards of OMG (2015) and is compatible with Camunda.

#### 4.6. Evaluation

The models’ performance is first assessed independently, followed by an evaluation of the complete DMN model generation using a test set of 36 DMN models.

**4.6.1. Object and Keypoint Detection Evaluation Metrics** Average precision (AP) is calculated for the object detection and keypoint detection models as follows with  $TP$  = True Positive and  $FP$  = False Positive:

$$AP = \frac{TP \text{ predictions}}{TP \text{ predictions} + FP \text{ predictions}}$$

Average recall (AR) and F1-score (F1) are not supported by the COCO evaluation package (“COCO: Common Objects in Context”, n.d.) and are thus not reported. However, AP is a good and common evaluation metric for this purpose. The tested DMN models only contain entirely new scenarios, ensuring the test set contains unique instances and further tests the robustness of this approach.

**4.6.2. Text Detection Evaluation Metrics** AP, AR and F1 are calculated to assess the performance of Tesseract OCR (Smith, 2007), by comparing the text predictions to manually constructed ground truth text files. The metrics were obtained as follow:

$$\text{Precision} = \frac{|TP|}{|\text{Predictions}|}$$

$$\text{Recall} = \frac{|TP|}{|\text{Ground Truth}|}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Levenshtein distances with a threshold of [1,2,3] are considered, measuring the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one word into another. This threshold was deemed acceptable as the DMN model can manually be edited later on by the user as long as the words are interpretable which with a Levenshtein of maximum 3 is acceptable.

**4.6.3. Complete DMN Evaluation Metrics** To evaluate a complete DMN model, each predicted element and arrow of a DRD is evaluated and each element of a decision table is evaluated. The following metrics are used with *pred* whether it was predicted by the model, *ok* indicating a correct prediction and *base* indicating whether it is in the ground truth:

$$\text{Precision: } \frac{\#elements_{ok}}{\#elements_{pred}}, \frac{\#table.elements_{ok}}{\#table.elements_{pred}} \text{ and } \frac{\#requirements_{ok}}{\#requirements_{pred}}$$

$$\text{Recall: } \frac{\#elements_{ok}}{\#elements_{base}}, \frac{\#table.elements_{ok}}{\#table.elements_{base}} \text{ and } \frac{\#requirements_{ok}}{\#requirements_{base}}$$

• F1-score: Harmonic mean of precision and recall

Lastly, each decision table is checked to be linked to the right decision. If the header of the input image corresponds to the output image’s header, the computed metrics of that pair of images are given a coefficient of 1. Otherwise, the computed metrics are multiplied by the following coefficient.

$$\bullet 1 - \frac{1}{\#decision\ elements\ predicted}$$

Then, the AP, AR and the average F1-score are calculated using the corresponding coefficients.

## 5. Experiments and Results

This section presents the results of our experiments. The results of the different models for the decision requirements level and decision logic level are reported in section 5.1 and section 5.2 respectively. Then, section 5.3 presents the results of the tool on the full DMN model generation examples.

As a reminder, the test set comprises complete DMN images, both digitally drawn and hand-drawn pairs, and each containing both a DRD and one corresponding decision table. These examples have not been utilized during the model training phase, showcasing the performance on entirely novel instances.

### 5.1. Results for DRDs

**5.1.1. Object Detection** Table 2 presents the AP obtained from the object detection models across different elements of DRDs, including decisions, input data, business knowledge, and knowledge sources. As expected, the highest AP is observed for decision elements in both datasets, due to the larger number of decision elements present in both datasets. Notably, APs exceed 0.8, demonstrating the effectiveness of the Faster R-CNN model (ShaoqingRen, 2015) in accurately detecting various elements for digitally drawn DRDs. Although the results for the hand-drawn dataset are less robust, they remain satisfactory, indicating potential for using object detection methods in handwritten contexts.

**Table 2. Results Object Detection Model-DRD**

Elements	Digital	Hand-drawn
	AP	
Decision	0.91	0.71
Input Data	0.86	0.68
Business Knowledge	0.87	0.32
Knowledge Source	0.81	0.41

**5.1.2. Keypoint Detection** Table 3 presents the AP obtained from the keypoint detection model across different requirements within DRDs, including information requirement, knowledge requirement and authority requirement. Overall, these results, exceeding 0.5, underscore the models effectiveness in detecting the head and tail of the different requirements, as

well as predicting their associated types. The highest AP is observed for the information requirement type in both datasets, as expected due to their larger presence. Interestingly, the highest AP for knowledge requirements is observed in the hand-drawn dataset, likely because human drawings distinguish arrow types more clearly than digital images.

**Table 3. Results Keypoint Detection Models-DRD**

Elements	Digital	Hand-drawn
	AP	
Information Requirement	0.90	0.90
Knowledge Requirement	0.49	0.55
Authority Requirement	0.68	0.49

**5.1.3. Text Detection** Table 4 presents the Average Precision (AP), Average Recall (AR), and F1-score achieved by the text detection model on both datasets. The metrics are calculated considering a Levenshtein distance of 3, which is deemed small enough for human recognition of words. By allowing for 3 variations in the characters, all metrics exceed 0.50.

**Table 4. Results Text Detection Model-DRD**

Metric	Digital	Hand-drawn
Average Precision	0.90	0.74
Average Recall	0.93	0.52
F1-score	0.90	0.60

### 5.2. Results for Decision Tables

**5.2.1. Table Structure Detection** Table 5 compares the AP of the elements predictions of the Faster R-CNN model (ShaoqingRen, 2015) and the Mask R-CNN model (matterport, 2018) for both the digitally drawn and hand-drawn dataset. The table focuses on table structure detection, giving results for each element in a decision table image, including decision/header, rule, hit policy, input, output, input entry and output entry. The results contained in the table suggest that the pre-trained Mask R-CNN model (matterport, 2018) is better performing for digitally drawn DMN images, while for hand-drawn DMN images the pre-trained Faster R-CNN model (ShaoqingRen, 2015) seems to predict more accurately.

**5.2.2. Text Detection** Table 6 shows the obtained metrics of the text detection models, following the same approach as for DRDs. Allowing for 3 variations in the characters showcase satisfactory results with the metrics exceeding 0.50.

**Table 5. Results Objects Detection Model-Logic**

Elements	Digital		Hand-drawn	
	Faster R-CNN	Masked R-CNN	F R-CNN	M R-CNN
	AP			
Decision/ header	0.72	<b>0.72</b>	<b>0.38</b>	0.43
Hit policy	0.77	<b>0.77</b>	<b>0.11</b>	0.13
Rule	0.45	<b>0.54</b>	<b>0.32</b>	0.29
Input	0.66	<b>0.67</b>	<b>0.41</b>	0.38
Output	0.70	<b>0.71</b>	<b>0.46</b>	0.43
Input entry	0.29	<b>0.27</b>	<b>0.29</b>	0.28
Output entry	0.32	<b>0.36</b>	<b>0.30</b>	0.28
Table	0.86	<b>0.85</b>	<b>0.56</b>	0.59

**Table 6. Results Text Detection Model-Logic**

Metric	Digital	Hand-drawn
AP	0.84	0.74
AR	0.81	0.52
F1	0.82	0.61

### 5.3. Results for complete DMN models

Table 7 showcases the comprehensive evaluation metrics for digitally drawn DMN images, with a Levenshtein distance threshold of maximum of 3 characters. Notably, all DRD metrics surpass a value of 0.70, showcasing effective DRD model prediction by the tool. Considering the total metrics, encompassing both DRD's elements and Table's elements, the metrics consistently exceed 0.65. Such performance metrics highlight the potential of the tool in practical real-world applications for digitally drawn contexts.

Similarly, Table 7 presents the metrics obtained for the hand-drawn images. The results indicate that the performance of predictions on hand-drawn images is comparatively lower. A significant challenge arises from the accurate detection of text, influencing the overall results and the table elements results in particular. Despite the relatively low results, the tool demonstrates potential in handling hand-drawn images.

## 6. Discussion, Limitations & Future Work

For DRDs, different objects and keypoints detection models show promising performance with both digitally drawn and hand-drawn DMN images, as shown in the results tables 2 and 3. Employing a pre-trained Faster R-CNN model for the digitally drawn DMN dataset achieves satisfactory performance despite a limited dataset. However, augmenting the dataset further with more, and newer DRD elements introduced by OMG (2015) could be beneficial. For the hand-drawn dataset, introducing additional styles of handwriting through augmentation would contribute to making the model more generalized. In general for both datasets, an augmentation could allow to create a validation set, and enable to fine-tune the model with optimal parameters.

It is worth noting that even though the collected DRD images show the DRDs in a top-down fashion, the pattern-based approach should be also be able to deal with any type of position of the DRD, but this should be further tested.

When considering text recognition models, the achieved results (see table 4) are satisfactory. However, accurately detecting text, especially handwritten text, remains challenging. In particular, the text detection is highly dependent on the image quality. To enhance performance, training a customized text recognition model using an expanded dataset could be beneficial.

At the decision logic level, employing pre-trained object detection models for recognizing various elements within a table is an innovative approach. The Mask R-CNN model's results, presented in table 5, indicate its effectiveness with digitally drawn decision tables. However, augmenting the dataset is essential to yield superior results. Additionally, although pre-trained models, especially a Faster R-CNN model, also demonstrated promising results with hand-drawn images (see table 5), sensitivity to table formats and diverging tables from the standard DMN style becomes more apparent. Therefore, exploring alternative table detection techniques could be further investigated. Future work could for example focus on leveraging the relative position of elements within the table once the table is identified, facilitating cell label prediction. Moreover, a more comprehensive post-processing pipeline could improve the results.

Regarding text detection, similar conclusion as those observed for the DRDs can be drawn.

The outcomes from the experiments for the full DMN model generation reveal that the tool serves as a solid prototype. While the results presented in 7 demonstrate promising performance, the tool's outcomes are dependent on the initial image's quality. Next, the evaluation was carried out on a reasonable test set of DMN models, potentially not capturing all possible DMN models and variations. However, the tool can be used as a first attempt to create a DMN file after first sketch was drawn after a meeting or based on a digital picture of a DMN model missing its source files.

In the future, it can be investigated how domain knowledge can be integrated with the pipeline so that the quality of the models can be improved on a semantic level for example. Even though currently the accuracy is not impressive, the approach can still be improved by adding more training data from the SAP-SAM dataset published by Sola et al. (2022) which will be manually annotated as future work. Next, tool's capabilities could be further expanded to allow the incorporation of more than one table at a time, ultimately enhancing user

**Table 7. Results Tool**

Metric	DRD Elements	DRD Arrows	Table Elements	Total
<b>Digital</b>				
AP	<b>0.86</b>	0.72	0.36	<b>0.65</b>
AR	<b>0.94</b>	0.87	0.51	<b>0.77</b>
Average F1-score	<b>0.90</b>	0.80	0.43	<b>0.71</b>
<b>Hand-drawn</b>				
AP	0.26	<b>0.61</b>	0.07	<b>0.32</b>
AR	0.29	<b>0.71</b>	0.31	<b>0.44</b>
Average F1-score	0.28	<b>0.66</b>	0.19	<b>0.38</b>

experience and utility.

The entirety of the DMN extraction pipeline detailed in this research is seamlessly integrated within the DMNComputerVisionTool, an open-source application. The tool and a video demonstrating its use are accessible through the following links: DMNComputerVisionTool, Demonstration Video.

## 7. Conclusion

In this study, deep learning techniques are used to extract information from DMN models depicted in both digitally drawn and hand-drawn images. A dataset was collected and annotated for the training and evaluation of various models. Pre-trained models leveraging the Detectron2 framework to detect DMN elements and leveraging Tesseract for text detection were used. Leveraging these models, a DMN extraction pipeline is proposed and implemented within an application, DMNComputerVisionTool allowing for a faster communication with stakeholders with DMN drafts. While there is room for improvement, the research findings showcase the feasibility of employing deep learning models to accurately extract DMN components. The datasets and different models are available on Hugging Face.

## References

Antinori, A., Coltrinari, R., Corradini, F., Fornari, F., Re, B., & Scarpetta, M. (2022). Bpmn-redrawer: From images to bpmn models. *BPM (PhD/Demos)*, 107–111.

Batoulis, K., Meyer, A., Bazhenova, E., Decker, G., & Weske, M. (2015). Extracting decision logic from process models. *CAiSE 2015, Stockholm, Sweden, June 8-12, 2015, Proceedings 27*, 349–366.

Bazhenova, E., Buelow, S., & Weske, M. (2016). Discovering decision models from event logs. *Business Information Systems: 19th International Conference, BIS 2016, Leipzig,*

*Germany, July, 6-8, 2016, Proceedings 19*, 237–251.

Bazhenova, E., Zerbato, F., Oliboni, B., & Weske, M. (2019). From bpmn process models to dmn decision models. *Information Systems*, 83, 69–88.

Biard, T., Le Mauff, A., Bigand, M., & Bourey, J.-P. (2015). Separation of decision modeling from business process modeling using new “decision model and notation”(dmn) for automating operational decision-making. *Risks and Resilience of Collaborative Networks: 16th IFIP WG 5.5 Working Conference on Virtual Enterprises, PRO-VE 2015, Albi, France,, October 5-7, 2015, Proceedings 16*, 489–496.

Coco: *Common objects in context*. (n.d.). <https://cocodataset.org/#format-data>

Corea, C., Kampik, T., & Delfmann, P. (2023). Empirical evidence of dmn errors in the wild-an sap signavio case study. *International Conference on Business Process Management*, 326–336.

dataTorch. (2024). *Datatorch*. <https://datatorch.io/>

De Smedt, J., Hasić, F., vanden Broucke, S. K., & Vanthienen, J. (2019). Holistic discovery of decision models from process execution data. *Knowledge-Based Systems*, 183, 104866.

Etikala, V., Van Veldhoven, Z., & Vanthienen, J. (2020). Text2dec: Extracting decision dependencies from natural language text for automated dmn decision modelling. *BPM*, 367–379.

Gantayat, N., Sridhara, G., Sankaran, A., Dechu, S., Mani, S., & Dasgupta, G. B. (2018). Towards creating business process models from images. *Service-Oriented Computing: 16th International Conference, ICSOC 2018, Hangzhou, China, November 12-15, 2018, Proceedings 16*, 100–108.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation.

- Proceedings of the IEEE conference on computer vision and pattern recognition*, 580–587.
- Goossens, A., De Smedt, J., & Vanthienen, J. (2023a). Comparing the performance of gpt-3 with bert for decision requirements modeling. *International Conference on Cooperative Information Systems*, 448–458.
- Goossens, A., De Smedt, J., & Vanthienen, J. (2023b). Extracting decision model and notation models from text using deep learning techniques. *Expert Systems with Applications*, 211, 118667.
- Goossens, A., De Smedt, J., & Vanthienen, J. (2024). Extracting process-aware decision models from object-centric process data. *arXiv preprint arXiv:2401.14847*.
- Goossens, A., Vandevelde, S., Vanthienen, J., & Vennekens, J. (2023). Gpt-3 for decision logic modeling. *Proceedings of the 17th International Rule Challenge and 7th Doctoral Consortium@ RuleML+ RR 2023 co-located with 19th Reasoning Web Summer School (RW 2023) and 15th DecisionCAMP 2023 as part of Declarative AI 2023*, 3485, 1–14.
- Kaiming, H., Xiangyu, Z., Shaoqing, R., & Jian, S. (2015). *Deep residual learning for image recognition*. <https://arxiv.org/abs/1512.03385>
- Karasneh, B., & Chaudron, M. R. (2013). Img2uml: A system for extracting uml models from images. *2013 39th Euromicro Conference on Software Engineering and Advanced Applications*, 134–137.
- matterport. (2018). *Mask r-cnn for object detection and segmentation*. [https://github.com/matterport/Mask\\_RCNN](https://github.com/matterport/Mask_RCNN)
- OMG. (2015). Omg: Decision model and notation 1.0 (2015) [Accessed: 2022-01-08]. <https://www.omg.org/spec/DMN/1.0/>
- Opencv. (2023). <https://github.com/opencv/opencv>
- Pienikazek, P. (2014). Automatic extraction of business logic from digital documents. In *Image processing & communications challenges 6* (pp. 161–168). Springer.
- Post, R., Smit, K., & Zoet, M. (2020). Adoption and implementation of the decision model and notation standard. *Journal of Advanced Management Science Vol*, 8(2).
- Quishpi, L., Carmona, J., & Padró, L. (2021). Extracting decision models from textual descriptions of processes. *BPM*, 85–102.
- Rozinat, A., & van der Aalst, W. M. (2006). Decision mining in prom. *Business Process Management: 4th International Conference, BPM 2006, Vienna, Austria, September 5-7, 2006. Proceedings 4*, 420–425.
- Sachin Raja, C. J., Ajoy Mondal. (2022). *Visual understanding of complex table structures from document images*. <https://shorturl.at/9B1rA>
- Schäfer, B., van der Aa, H., Leopold, H., & Stuckenschmidt, H. (2021). Sketch2bpmn: Automatic recognition of hand-drawn bpmn models. *International Conference on Advanced Information Systems Engineering*, 344–360.
- Schäfer, B., van der Aa, H., Leopold, H., & Stuckenschmidt, H. (2022). Sketch2process: End-to-end bpmn sketch recognition based on neural networks. *IEEE Transactions on Software Engineering*.
- ShaoqingRen. (2015). *Faster r-cnn: Towards real-time object detection with region proposal networks*. [https://github.com/ShaoqingRen/faster\\_rcnn](https://github.com/ShaoqingRen/faster_rcnn)
- Singh, A., Bacchuwar, K., & Bhasin, A. (2012). A survey of ocr applications. *International Journal of Machine Learning and Computing*, 2(3), 314.
- Smith, R. (2007). An overview of the tesseract ocr engine. *Ninth international conference on document analysis and recognition (ICDAR 2007)*, 2, 629–633.
- Sola, D., Warmuth, C., Schäfer, B., Badakhshan, P., Rehse, J.-R., & Kampik, T. (2022). Sap signavio academic models: A large process model dataset. *International Conference on Process Mining*, 453–465.
- Tsung-Yi, L., Piotr, D., Ross, G., Kaiming, H., Bharath, H., & Belongie, S. (2016). *Feature pyramid networks for object detection*. <https://arxiv.org/abs/1612.03144>
- Vanthienen, J. (2021). Decisions, advice and explanation: An overview and research agenda. *A Research Agenda for Knowledge Management and Analytics*, 149–169.
- Vanthienen, J., & Goossens, A. (2024). From text to intelligent services in knowledge intensive decision processes: Text2chat. <https://scholarspace.manoa.hawaii.edu/items/81b3ea6f-9f81-4648-904d-6f239d6c4776>
- Vanthienen, J., Mues, C., & Aerts, A. (1998). An illustration of verification and validation in the modelling phase of kbs development. *Data & Knowledge Engineering*, 27(3), 337–352.
- Wu, Y., Kirillov, A., Massa, F., Lo, W.-Y., & Girshick, R. (2019). Detectron2.