A COMPARATIVE STUDY OF DEEP LEARNING MODELS FOR SYMBOL DETECTION IN TECHNICAL DRAWINGS

Benedikt Faltin, Damaris Gann & Markus König
Ruhr University Bochum, Germany

ABSTRACT: *Symbols are a universal way to convey complex information in technical drawings since they can represent a wide range of elements, including components, materials, or relationships, in a concise and space-saving manner. Therefore, to enable a digital and automatic interpretation of pixel-based drawings, accurate detection of symbols is a crucial step. To enhance the efficiency of the digitization process, current research focuses on automating this symbol detection using deep learning models. However, the ever-increasing repertoire of model architectures poses a challenge for researchers and practitioners alike in retaining an overview of the latest advancements and selecting the most suitable model architecture for their respective use cases. To provide guidance, this contribution conducts a comparative study of prevalent and state-of-the-art model architectures for the task of symbol detection in pixel-based construction drawings. Therefore, this study evaluates six different object detection model architectures, including YOLOv5, YOLOv7, YOLOv8, Swin-Transformer, ConvNeXt, and Faster-RCNN. These models are trained and tested on two distinct datasets from the bridge and residential building domains, both representing substantial sub-sectors of the construction industry. Furthermore, the models are evaluated based on five criteria, i.e., detection accuracy, robustness to data scarcity, training time, inference time, and model size. In summary, our comparative study highlights the performance and capabilities of different deep learning models for symbol detection in construction drawings. Through the comprehensive evaluation and practical insights, this research facilitates the advancement of automated symbol detection by showing the strengths and weaknesses of the model architectures, thus providing users with valuable guidance in choosing the most appropriate model for their real-world applications.*

KEYWORDS: *Computer Vision, Technical Drawings, Symbol Detection, Comparative Study*

1. INTRODUCTION

Symbols pose an efficient and space-saving way of conveying complex information, enabling understanding across languages due to their standardized appearance. They find application in diverse contexts, such as street signs (Gudigar et al., 2016), maps (Huang et al., 2023), or technical drawings (Elyan et al., 2020b). For instance, in construction drawings symbols can represent architectural components, plumbing fixtures, elevation markings, and more, making accurate identification of the symbols essential for understanding the entire drawing. The importance of precise symbol detection becomes even more apparent when algorithms are used to understand the technical drawings automatically, e.g., for their digitization. Therefore, research has focused on developing effective and accurate algorithms for locating and classifying symbols in technical drawings. Ah-Soon (1998) proposed to adapt Messmer's algorithm for symbol recognition in architectural drawings using graph matching. The drawing is first vectorized, followed by the extraction and merging of geometric features for each symbol, and clustering similar symbols by type. In a separate work, Adam et al. (2000) developed an orientation and scale invariant method for recognizing symbols in technical documents. Their algorithm is based on the Fourier-Mellin transform, which extracts features used to label the symbols through a classifier.

Most of this research focuses on traditional image analysis techniques, such as vectorization and feature engineering. However, with the advent of efficient deep learning approaches, such as convolutional neural networks (CNN), the research community's interest has shifted to use such models for localizing and classifying symbols in technical drawings. Ziran and Marinai (2018) leverages the object detection networks Faster R-CNN and Single Shot Detector (SSD) to detect symbols representing objects such as furniture, doors, and windows in architectural floor plans. In the context of piping and instrumentation diagrams (P&IDs), Mani et al. (2020) develops a custom CNN to classify components in the P&IDs. The proposed network is trained to classify patches cropped from the overall drawing into three classes: tag symbol, component symbol, or no symbol. When the network detects a symbol within a patch, the corresponding detection is projected back onto the original drawing. In a different approach for P&IDs, Elyan et al. (2020a) employs the YOLO architecture for symbol detection. Additionally, the authors propose a method based on generative adversarial networks to mitigate the issue of class imbalance in technical drawings. Class imbalance occurs when the number of instances per symbol class shows significant variation. A novel approach to enhance symbol detection by inferring the symbol orientation is

Referee List (DOI: 10.36253/fup_referee_list)
FUP Best Practice in Scholarly Publishing (DOI 10.36253/fup_best_practice)
Benedikt Faltin, Damaris Gann, Markus König, *A Comparative Study of Deep Learning Models for Symbol Detection in Technical Drawings,* pp. 877-886, © 2023 Author(s), CC BY NC 4.0, DOI 10.36253/979-12-215-0289-3.87

introduced by Faltin et al. (2023b). The authors leverage human pose detection networks, specifically Mask R-CNN and YOLOv7Pose, to achieve accurate symbol pose estimation in construction drawings.

While the proposed approaches already utilize several different detection models, the overall field of object detection has experienced rapid growth, leading to the emergence of numerous model architectures (Zaidi et al., 2022). Researchers and industry practitioners alike face the difficult task of selecting the most appropriate architecture for their specific applications. This becomes especially difficult when other critical requirements such as training time, inference time, or model size must be considered in addition to detection accuracy. For instance, in real-time applications, faster inference time may be prioritized, while in resource-constrained environments, smaller model sizes may be important. Therefore, researchers and practitioners need to conduct thorough evaluations of the various object detection models to make an informed decision.

Previous research has compared different model architectures for different applications to aid in model selection. However, to the best of the authors' knowledge, there is no study that directly compares detection models for symbol detection in pixel-based drawings. Nevertheless, related publications have conducted comparison studies in various other domains. For instance, Brößner et al. (2022) compares nnUNet with the transformer-based Swin-UNet for bone segmentation in ultrasound images, finding that both networks are similarly applicable to the task. In another study, Wang et al. (2023) conducts a comparison for different backbones such as ResNet, Swin, and ViTAEv2, by pre-training them on a large dataset of satellite images to later test them on different down-stream tasks, such as image segmentation or object detection. The authors discover that the transformer-based models show competitive results compared with the CNN models. In particular, ViTAEv2 achieves the highest performance on the different tasks. Moutik et al. (2023) compares several models based on CNNs, transformers, and hybrid approaches, to recognize human actions in video data. This study concludes that the CNN- and transformer-based models perform similarly, both with their strengths and weaknesses, but overall, the hybrid methods achieve the best results.

Our research contributes to this broad research area by providing a comprehensive analysis of state-of-the-art object detection models and evaluating them based on several criteria for symbol detection in construction drawings. The remainder of the paper is structured as follows: Section 2 presents the methodological design of the comparative study conducted and details of the datasets used. Subsequently, in Section 3, we present and discuss the results of our comparative study, shedding light on the performance of different detection models. Finally, Section 4 summarizes the results and provides possible directions for further research in this area.

2. METHODOLOGY

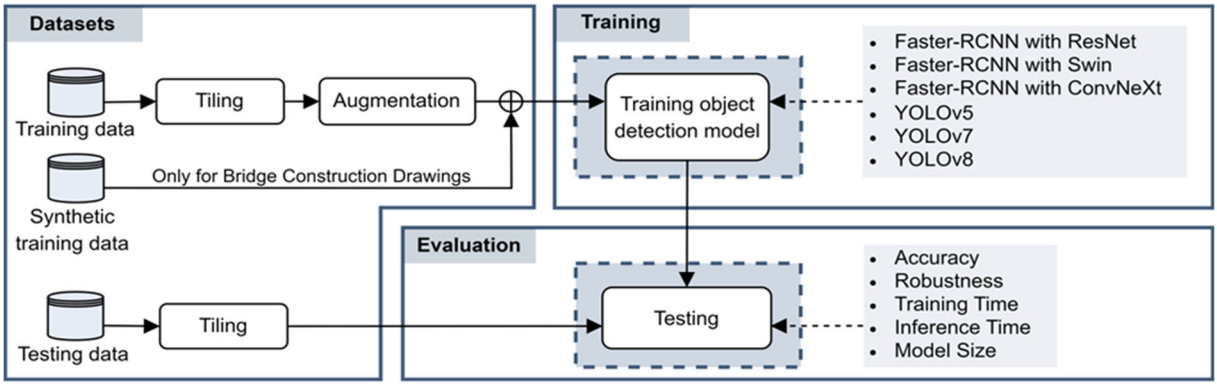

Fig. 1: Overview of the methodology used in the comparative study.

The structure of this comparative study is shown in Fig. 1. Real data is collected and annotated to train and test the selected models. In addition, synthetic data is generated to extend the dataset. The models are trained with a combination of real and synthetic data, while only real data is used for testing. Section 2.1 provides a detailed description of the datasets. The comparison includes six different network architectures: YOLOv5 (Jocher et al., 2022), YOLOv7 (Wang et al., 2022), YOLOv8 (Jocher et al., 2023), and Faster R-CNN (Ren et al., 2017) with three different backbones namely ResNet (He et al., 2016), Swin (Liu et al., 2021), and ConvNeXt (Liu et al., 2022). A brief introduction to the compared network architectures is presented in Section 2.2. In order to comprehensively assess the models, they are evaluated with respect to five criteria, i.e., accuracy, robustness, training time, inference time, and model size, which are explained in detail in Section 2.3.

2.1 Datasets

To obtain a meaningful comparison, the models are trained and tested on two distinct datasets representing different sub-sectors of the construction industry: bridge construction drawings and floor plans. The *bridge*-dataset comprises 15 real construction drawings with approximately 10.000 x 6.000 pixels dimensions. To make the large dimensions of the drawings manageable for the models, tiling is used to divide the drawings into patches. The drawings contain three types of symbols: section symbols, elevation markers, and dimension symbols (cf. Fig. 2). The size of these symbols varies from 20 to 300 pixels, which is relatively small compared to the overall size of the drawing. Consequently, the models must be able to produce meaningful feature representations even for small objects. To investigate this, the study explores different patch sizes to examine the influence of the symbol-to-image size ratio. Therefore, three patch sizes are considered: 256 pixels, 512 pixels, and 1024 pixels. This results in three data subsets named *bridge_256*, *bridge_512*, and *bridge_1024*, containing 2120, 1185, and 543 patches, respectively.

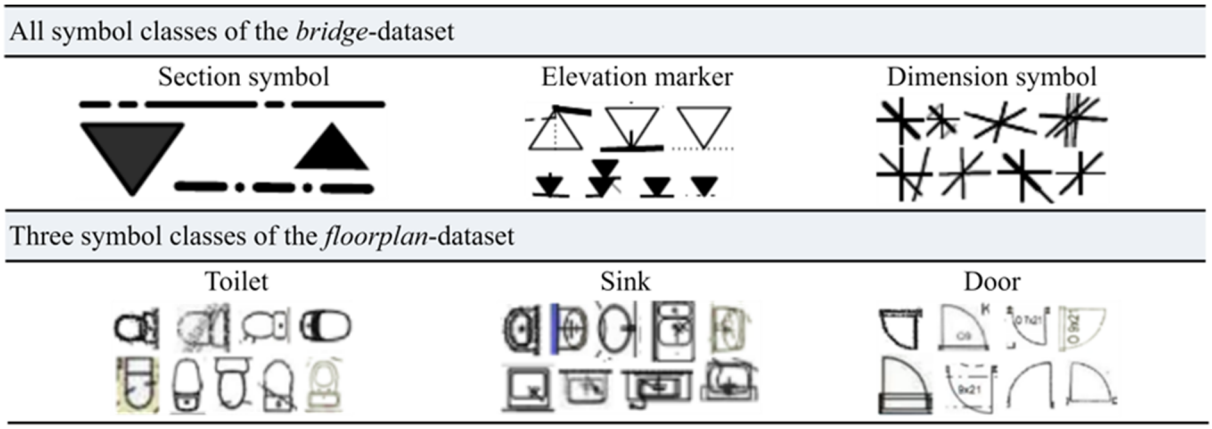

Fig. 2: Example illustration of some representative symbols selected from the datasets.

Since the number of patches is still quite limited, additional synthetic data is generated based on the procedure introduced by Faltin et al. (2022a). Lastly, to investigate the influence of the quantity of training data on the model's performance for the *bridge*-dataset, a reduced data subset called *bridge_1024_red* is created, which comprises only 10% of the original data subset *bridge_1024*. Table 1 gives an overview of the composition of all datasets.

Table 1: Overview of the composition of the data subsets. Synthetic data is only used for training and validation.

	No. training images	No. synthetic training images	No. validation images	No. synthetic validation images	No. testing images
bridge_256	5611	1500	993	250	316
bridge_512	3179	1500	544	250	192
bridge_1024	1477	1500	250	250	94
bridge_1024_red	130	170	25	25	94
floorplan_1024	10294	0	1800	0	750

On the other hand, the *floorplan*-dataset consists of 5000 fully annotated floor plans of residential buildings sourced from the CubiCasa-dataset (Kalervo et al., 2019). The majority of the drawings in this dataset are similarly sized ranging between 500 and 2000 pixels, therefore they are resized uniformly to 1024 pixels (*floorplan_1024*). The drawings distinguish eight different symbol classes: window, door, electrical appliance, toilet, sink, sauna bench, fireplace, and bathtub. Fig. 2 illustrates the high intra-class variation among these symbols, which requires effective generalization of the object detection models to handle the diverse styles.

2.2 Object Detection Models

In this study three single-stage detection models, namely YOLOv5, YOLOv7, and YOLOv8 are compared with three two-stage detection models. Each two-stage model modifies the Faster R-CNN architecture by replacing the backbone while keeping the region proposal network and detection head unaltered. The selected backbones are ResNet, ConvNeXt, and Swin. In this section, a short overview of the different architectures is given.

The YOLO model series is known for its high detection performance and fast inference speed. Both YOLOv7 and YOLOv8 are extensions of YOLOv5 but incorporate different ideas such as anchorless detection and different convolutional blocks to enhance the base model's performance.

In comparison, ResNet is a well-established CNN-based image classification network commonly used as the backbone architecture for Faster R-CNN. On the other hand, the Swin backbone is a transformer-based architecture, building upon the concept of the visual transformer introduced by Dosovitskiy et al. (2020). Swin addresses the challenges of adapting transformers to the image domain by employing a hierarchical architecture with shifting attention windows. Lastly, ConvNeXt progressively modernizes the basic ResNet architecture by incorporating key components of visual transformers into the CNN-based architecture.

For each of the chosen model architectures, the base size and the smallest version, referred to as tiny, are compared. This investigates whether equivalent results can be obtained with smaller model sizes. In general, larger models can extract more meaningful features due to their increased complexity, allowing them to handle challenging tasks while, in contrast, making them prone to overfit. The smallest versions are ResNet-50, Swin-Tiny, ConvNeXt-Tiny, YOLOv5n, YOLOv7-tiny, and YOLOv8n, while the base-versions are ResNet-101, Swin-Base, ConvNeXt-Base, YOLOv5m, YOLOv7, and YOLOv8m.

2.3 Evaluation

To provide a comprehensive overview of the strengths and weaknesses of each model, they are evaluated based on five criteria: accuracy, robustness, training time, inference time, and model size. A detailed explanation of each criterium is given in the remainder of this section:

2.3.1 Accuracy

Accuracy measures the model's ability to correctly locate and classify the symbols within the drawing. To assess the accuracy the standard metric of mean average precision (mAP) as defined by Padilla et al. (2021) is utilized, as it considers the precision and recall, evaluated at certain levels of intersection over union (IoU). While the precision indicates the proportion of accurate predictions made by the model among all the predictions, the recall denotes the proportion of correct predictions compared to the total number of ground-truth boxes available. Lastly, the IoU quantifies the overlap between a prediction and the ground-truth bounding box, measuring the model's localization accuracy.

2.3.2 Robustness

Obtaining and annotating extensive training data is time-consuming, resulting in data scarcity in many cases. Hence, it is highly desirable to have a robust object detection network, which means it performs reasonably well even with limited training data. The model is trained with 90% reduced training data to evaluate robustness. After training, the network's performance is tested with the full test dataset using the mAP metric. This enables an estimate of the accuracy decrease when training data is significantly limited.

2.3.3 Training time

Training time measures the physical time required for the model to reach a plateau, indicating that no further improvement is expected. This is particularly important in applications with limited computational resources or where the model is regularly retrained. While the training time may vary depending on the hardware used, in this study all models are trained on the same GPU to ensure comparability.

2.3.4 Inference Time

The inference time quantifies how long it takes for the model to make a single prediction, making it a vital metric for real-time applications.

2.3.5 Model size

Model size describes the memory requirements of the model and is a crucial factor that directly affects various aspects of model performance. Smaller models usually have a smaller number of parameters, leading to shorter inference times. However, extremely small models may not have the necessary complexity to handle the given task effectively. Therefore, it is important to carefully select the appropriate model size to achieve optimal performance.

3. RESULTS & DISCUSSION

To perform the comparative study, the models are trained by running three Nvidia A100-SXM4-40GB GPUs. The two-level networks are available pre-trained solely on the ImageNet-1K dataset, which is a subset of the larger ImageNet dataset (Deng et al., 2009). Conversely, the YOLO models are available pre-trained exclusively on the MS COCO dataset (Lin et al., 2014). The AdamW (Loshchilov & Hutter, 2017) optimizer is employed with a batch size of 64 and an initial learning rate of 0.0001 for backpropagation. The maximum number of epochs is set to 400, but early stopping is utilized to prevent overfitting. In order to increase the overall quantity and diversity of the training data, data augmentation techniques such as flipping, rotation, scale, or translation are utilized for all data subsets. The results are presented and discussed in the following sections based on the different evaluation criteria.

3.1 Results for Accuracy

Table 2 shows the results of the comparative study focusing on symbol recognition accuracy. To evaluate the symbol detection performance with the mAP metric, an IoU threshold of 0.5 (mAP@0.50) is appropriate for most use cases. Nevertheless, the more stringent mAP@0.50:0.95 is also used in this study to evaluate the models more comprehensively. For the bridge dataset, ConvNeXt-B achieves the highest mAP@0.50 score of 0.996 on 512 x 512 pixel patches, closely followed by ResNet-101, Swin-T, and Swin-B, all obtaining a score of 0.992 on 1024 x 1024 pixel patches. This suggests that the symbol-to-image size ratio is not critical, as the models accurately detect smaller symbols even on larger patches. But on the contrary, performance generally declines with smaller patch sizes, with the lowest results observed on 256 x 256 pixel patches. This demonstrates the importance of context, as the models must consider each symbol's surrounding space to locate and classify it properly. This is consistent with the observations of Lim et al. (2019), who similarly emphasized the importance of context when dealing with small objects.

Table 2: Performance results of the detection models on the *bridge*- and *floorplan*-dataset. The bold fonts indicate the best results for the respective data subset, while the asterisk indicates the best results for the specific dataset.

	Bridge-dataset						*Floorplan*-dataset	
	256 x 256 Pixel		512 x 512 Pixel		1024 x 1024 Pixel		1024 x 1024 Pixel	
	$mAP^{IoU}_{0.50}$	$mAP^{IoU}_{0.50:0.95}$	$mAP^{IoU}_{0.50}$	$mAP^{IoU}_{0.50:0.95}$	$mAP^{IoU}_{0.50}$	$mAP^{IoU}_{0.50:0.95}$	$mAP^{IoU}_{0.50}$	$mAP^{IoU}_{0.50:0.95}$
ResNet-50	0.912	0.739	0.973	0.815	0.986	0.833	0.771	0.492
ResNet-101	0.927	0.743	0.980	0.827	**0.992**	0.830	0.776	0.485
Swin-T	0.945	0.769	0.961	0.801	**0.992**	0.846	0.789	0.517
Swin-B	**0.950**	0.742	0.978	0.834	**0.992**	0.847	0.799	0.518
ConvNeXt-T	0.934	0.763	0.992	0.862	0.991	0.855	0.795	0.527
ConvNeXt-B	0.934	0.786	***0.996**	0.864	0.991	0.862	0.806	0.524
YOLOv5n	0.884	0.578	0.868	0.615	0.912	0.645	0.649	0.379
YOLOv5m	0.886	0.704	0.951	0.799	0.973	0.838	0.761	0.510
YOLOv7-tiny	0.884	0.648	0.924	0.675	0.949	0.686	0.763	0.474
YOLOv7	0.934	0.705	0.956	0.786	0.967	0.748	***0.816**	0.551
YOLOv8n	0.921	0.715	0.890	0.705	0.965	0.794	0.717	0.454
YOLOv8m	0.938	0.814	0.965	0.854	0.982	0.891	0.805	0.570

Regarding mAP@0.50:0.95, ConvNeXt-B still achieves the best results with 0.864. Conversely, YOLOv5n is the weakest performer in both mAP@0.50 and mAP@0.50:0.95, likely due to its lower model complexity affecting its ability to handle the complex symbol detection task.

For the *floorplan*-dataset, YOLOv7 leads by a significant margin (0.816), followed by ConvNeXt-B (0.806) and YOLOv8m (0.805). However, when considering mAP@0.50:0.95, YOLOv8m outperforms YOLOv7 with a notable gap. The general accuracy on the floor plans is lower compared to that of the bridge drawings. This discrepancy may be due to a larger number of symbol classes or a wider variety of symbol styles in the *floorplan*-dataset.

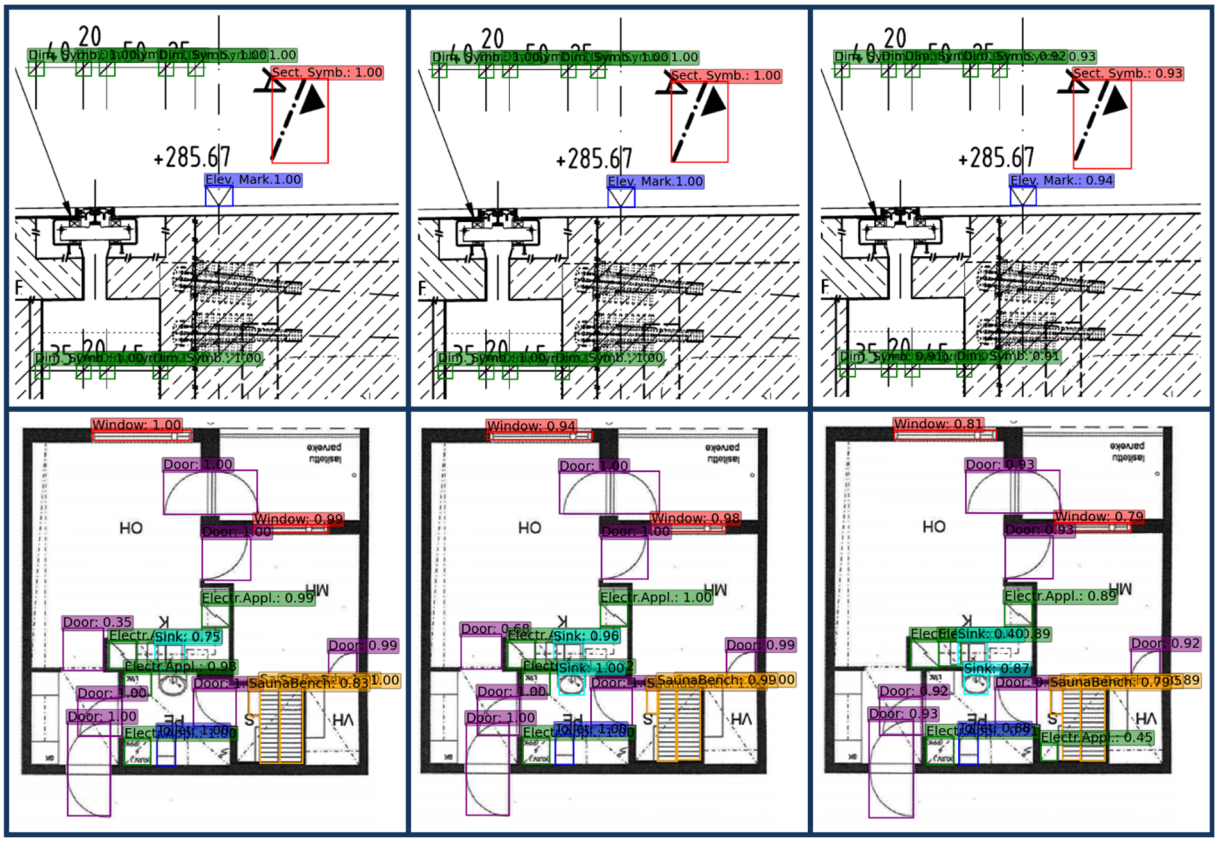

Fig. 3: Predictions of the base models of Swin (left), ConvNeXt (center), and YOLOv8 (right) on a patch from the *bridge*-dataset (top) and the *floorplan*-dataset (bottom).

Overall, larger and newer models perform better when accuracy matters. However, no significant difference in detection accuracy is seen between transformer-based and modern CNN-based networks. Despite that, the YOLO models tend to achieve lower mAP values than the two-stage models. This might be due to the YOLO model's smaller size. One exception is the YOLOv8m model, which achieves the highest values of all YOLO models and sometimes even outperforms the two-stage models.

3.2 Results for Robustness

The results for the reduced training dataset are presented in Table 3. While ConvNeXt-B shows the highest mAP score on the full training dataset, its performance declined by 25% when trained on the reduced dataset. It is also noteworthy that YOLOv5n, which already performed the worst, showed the largest decrease with 51%. On the other hand, YOLOv7 is quite robust to reduced training data, showing an 18% drop with mAP@0.50:0.95 score of 0.614. It is followed by the second most robust network ResNet-101 with a decrease of 26%.

3.3 Results for Training Time, Inference Time & Model Size

Table 3 shows the training time, inference time, and model size of the compared object recognition networks. Among them, YOLOv8n has the shortest training time with 0.51 hours on average, while still achieving promising results, as shown in Table 2. On the other hand, the transformer-based Swin-B network requires the longest training time. This is probably due to the computationally intensive attention mechanism used by the Swin network. In terms of inference time, the YOLO models demonstrate superior performance compared with the two-stage models. Specifically, YOLOv7-tiny achieves the fastest inference time, averaging 1.6 milliseconds per image, followed by YOLOv7 with 6.4 milliseconds per image. Surprisingly, despite its smaller network size, YOLOv5n does not

display the fastest inference time. It is worth noting that even the largest network, ConvNeXt-B, has a relatively small memory requirement of 400 MB.

Table 3: Performance results on the reduced *bridge*-dataset as well as training time, inference time, and model size. The bold fonts indicate the best results for the respective column.

	bridge_1024_red				
	$mAP_{0.50}^{IoU}$	Performance decrease	Training time in hrs. averaged	Inference in ms/img on 1024 x 1024 pixels	Model size in MB
ResNet-50	0.590	-31%	2.58	52.9	157.97
ResNet-101	0.611	-26%	3.21	55.3	230.62
Swin-T	0.635	-25%	5.19	70.4	170.95
Swin-B	0.602	-29%	7.52	109.9	397.56
ConvNeXt-T	0.633	-26%	4.20	64.5	171.88
ConvNeXt-B	0.645	-25%	5.51	104.2	400.27
YOLOv5n	0.319	-51%	0.82	12.9	**7.12**
YOLOv5m	0.541	-35%	2.67	17.8	80.77
YOLOv7-tiny	0.491	-28%	1.12	**1.6**	22.94
YOLOv7	**0.614**	**-18%**	3.15	6.4	139.21
YOLOv8n	0.570	-28%	**0.51**	12.9	11.49
YOLOv8m	0.672	-25%	2.49	18.3	98.65

In summary, it is important to find a balance between model size and accuracy. Smaller models may perform worse and models that are too large may overfit on the training data. Therefore, when choosing an appropriate model size, an optimal trade-off between model complexity and performance must be made based on the specific requirements of the application.

4. CONCLUSION

This paper provides a comprehensive comparison of six object detection models employed for the task of symbol detection in pixel-based construction drawings. The evaluated models include YOLOv5, YOLOv7, YOLOv8, and Faster R-CNN, equipped with three different backbones, i.e., ResNet, Swin, and ConvNeXt. For each architecture the smallest and baseline model size are employed, resulting in a total of twelve compared models. The evaluation is conducted along five key criteria, namely accuracy, robustness, training time, inference time, and model size. This offers valuable insights to guide the selection of appropriate models for diverse use cases and their specific requirements. The models are trained and tested on bridge construction drawings as well as floor plans representing two common sub-sectors of the construction industry.

Among the evaluated models, ConvNeXt shows the highest accuracy in the *bridge*-dataset, which makes it the best choice for detecting small symbols with low variance. On the other hand, when a high variance of symbol style is present, YOLOv7 or YOLOv8 are more suitable choices. Notably, YOLOv7 also proves robust when trained with reduced data, therefore making it superior to YOLOv8. Overall, the YOLO models generally have lower training and inference times, which can be directly linked to their smaller model size. Overall, the more recent models, including YOLOv8, Swin, and ConvNeXt, show comparably high accuracy, suggesting that significant improvements in the future will be limited. Nevertheless, promising research directions such as self-supervised learning (Jaiswal et al., 2021) and active learning (Schmidt et al., 2020) are expected to advance training efficiency by reducing the amount of training data required or the manual annotation effort. Moreover, expanding upon the proposal by Elyan et al. (2020a) for generating synthetic data through deep learning models, employing new methods such as stable diffusion (Rombach et al., 2022), could improve detection results even more, mitigating the scarcity of real data.

The results of our study provide valuable insights for the future development of symbol detection research, enabling further exploration of model performance on other technical drawing types, such as P&IDs. Investigating

hybrid models that combine transformer- and CNN-based architectures represents another promising direction for future research.

5. ACKNOWLEDGMENTS

This research is being conducted as part of the BIMKIT project, which is funded by the German Federal Ministry of Economics and Climate Protection. Furthermore, we would like to thank Lisa Freiin von Rössing for her valuable contributions to this study.

REFERENCES

Adam, S., Ogier, J. M., Cariou, C., Mullot, R., Labiche, J., & Gardes, J. (2000). Symbol and character recognition: application to engineering drawings. *International Journal on Document Analysis and Recognition*, *3*(2), 89–101. https://doi.org/10.1007/s100320000033

Ah-Soon, C. (1998). A constraint network for symbol detection in architectural drawings. In K. Tombre & A.K. Chhabra (Eds.), *Lecture Notes in Computer Science*. Springer. https://doi.org/10.1007/3-540-64381-8_41

Brößner, P., Hohlmann, B., & Radermacher, K. (2022). Transformer vs. CNN: A Comparison on Knee Segmentation in Ultrasound Images. In F. Rodriguez Y Baena, J. W. Giles & E. Stindel (Eds.), *Proceedings of the 20th Annual Meeting of the International Society for Computer Assisted Orthopaedic Surgery, Vol. 5,* 31–36. https://doi.org/10.29007/cqcv

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, & Li Fei-Fei (2009 ImageNet: A large-scale hierarchical image database. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,* 248–255. https://doi.org/10.1109/CVPR.2009.5206848

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2020). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. arXiv. https://doi.org/10.48550/arXiv.2010.11929

Elyan, E., Jamieson, L., & Ali-Gombe, A. (2020). Deep learning for symbols detection and classification in engineering drawings. *Neural networks*, *Vol. 129*, 91–102. https://doi.org/10.1016/j.neunet.2020.05.025

Elyan, E., Moreno-García, C. F., & Johnston, P. (2020). Symbols in Engineering Drawings (SiED): An Imbalanced Dataset Benchmarked by Convolutional Neural Networks. In L. Iliadis, P. P. Angelov, C. Jayne, & E. Pimenidis (Eds.), *Proceedings of the 21st EANN (Engineering Applications of Neural Networks) 2020 Conference,* 215–224. Springer. https://doi.org/10.1007/978-3-030-48791-1_16

Faltin, B., Schönfelder, P., & König, M. (2023). Inferring Interconnections of Construction Drawings for Bridges Using Deep Learning-based Methods. In E. Hjelseth, S. F. Sujan & R. J. Scherer (Eds.), *ECPPM 2022-eWork and eBusiness in Architecture, Engineering and Construction 2022*, 343-350. CRC Press. https://doi.org/10.1201/9781003354222

Faltin, B., Schönfelder, P., & König, M. (2023). Improving Symbol Detection on Engineering Drawings Using a Keypoint-Based Deep Learning Approach. *The 30th EG-ICE: International Conference on Intelligent Computing in Engineering.* https://www.ucl.ac.uk/bartlett/construction/sites/bartlett_construction/files/1889.pdf

Gudigar, A., Chokkadi, S., & U, R. (2016). A review on automatic detection and recognition of traffic sign. *Multimedia Tools and Applications*, *75*(1), 333–364. https://doi.org/10.1007/s11042-014-2293-7

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770-778. https://doi.org/10.1109/CVPR.2016.90

Huang, W., Sun, Q., Yu, A., Guo, W., Xu, Q., Wen, B., & Xu, L. (2023). Leveraging Deep Convolutional Neural Network for Point Symbol Recognition in Scanned Topographic Maps. *ISPRS International Journal of Geo-Information*, *12*(3), 128. https://doi.org/10.3390/ijgi12030128

Jaiswal, A., Babu, A. R., Zadeh, M. Z., Banerjee, D., & Makedon, F. (2021). A Survey on Contrastive Self-Supervised Learning. *Technologies*, *9*(1), Article 2. https://doi.org/10.3390/technologies9010002

Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., Kwon, Y., Michael, K., TaoXie, Fang, J. imyhxy, Lorna, Zan Yifu, Wong, C., V, A., Montes, D., Wang, Z., Fati, C., Nadar, J., Laughing, … Jain, M. (2022). ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation. Zenodo. https://doi.org/10.5281/zenodo.3908559

Jocher, G., Chaurasia, A., & Qiu, J. (2023). YOLO by Ultralytics (Version 8.0.0). https://github.com/ultralytics/ultralytics

Kalervo, A., Ylioinas, J., Häikiö, M., Karhu, A., & Kannala, J. (2019). CubiCasa5K: A Dataset and an Improved Multi-task Model for Floorplan Image Analysis. In M. Felsberg, P.-E. Forssén, I.-M. Sintorn & J. Unger (Eds.), *Image Analysis: 21st Scandinavian Conference*, *Vol. 11482,* 28-40. Springer. https://doi.org/10.1007/978-3-030-20205-7_3

Lin, T. Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., & Piotr, D. (2014). Microsoft COCO: Common Objects in Context. In D. Fleet, T. Pajdla, B.Schiele & T. Tuytelaars (Eds.), *Computer Vision – ECCV 2014*, *Vol. 13*, 740-755. Springer. https://doi.org/10.1007/978-3-319-10602-1_48

Lim, J.-S., Astrid, M., Yoon, H.-J., & Lee, S.-I. (2021). Small Object Detection using Context and Attention. *2021 International Conference on Artificial Intelligence in Information and Communication*, 181–186. https://doi.org/10.1109/ICAIIC51459.2021.9415217

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. *Proceedings of the IEEE/CVF International Conference on Computer Vision,* 9992-10002. https://doi.org/10.1109/ICCV48922.2021.00986

Liu, Z., Mao, H., Wu, C.Y., Feichtenhofer, C., Darrell, T., & Xie, S. (2022). A ConvNet for the 2020s. *Proceedings of the* IEEE *Conference on Computer Vision and Pattern Recognition,* 11976-11986. https://doi.org/10.1109/CVPR52688.2022.01167

Loshchilov, I., & Hutter, F. (2017). *Decoupled weight decay regularization*. arXiv. https://doi.org/10.48550/arXiv.1711.05101

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. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 176-177. https://doi.org/10.1109/CVPRW50498.2020.00096

Moutik, O., Sekkat, H., Tigani, S., Chehri, A., Saadane, R., Tchakoucht, T. A., & Paul, A. (2023). Convolutional Neural Networks or Vision Transformers: Who Will Win the Race for Action Recognitions in Visual Data?. *Sensors, 23*(2), 734. https://doi.org/10.3390/s23020734

Padilla, R., Passos, W. L., Dias, T. L. B., Netto, S. L., & da Silva, E. A. B. (2021). A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit. *Electronics*, *10*(3), 279. https://doi.org/10.3390/electronics10030279

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *Proceedings of the IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137-1149. https://doi.org/10.1109/TPAMI.2016.2577031

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-Resolution Image Synthesis with Latent Diffusion Models. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 10684-10695. https://doi.org/10.1109/CVPR52688.2022.01042

Schmidt, S., Rao, Q., Tatsch, J., & Knoll, A. (2020). Advanced Active Learning Strategies for Object Detection. *Proceedings of the IEEE Intelligent Vehicles Symposium.* 871–876. https://doi.org/10.1109/IV47402.2020.9304565

Wang, D., Zhang, J., Du, B., Xia, G. S., & Tao, D. (2023). An Empirical Study of Remote Sensing Pretraining. *Proceedings of the IEEE Transactions on Geoscience and Remote Sensing*, *61*. https://doi.org/10.1109/TGRS.2022.3176603

Wang, C.Y., Bochkovskiy, A., & Liao, H.Y. (2022). *YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors*. arXiv. https://doi.org/10.48550/arXiv.2207.02696

Zaidi, S. S. A., Ansari, M. S., Aslam, A., Kanwal, N., Asghar, M., & Lee, B. (2022). A survey of modern deep learning based object detection models. *Digital Signal Processing, 126*, Article 103514. https://doi.org/10.1016/j.dsp.2022.103514

Ziran, Z., & Marinai, S. (2018). Object Detection in Floor Plan Images. In: L. Pancioni, F. Schwenker, E. Trentin, (Eds.), *Artificial Neural Networks in Pattern Recognition,* 383-394. Springer. https://doi.org/10.1007/978-3-319-99978-4_30