10-9-2023

How Best to Hunt a Mammoth - Toward Automated Knowledge Extraction From Graphical Research Models

Sebastian Huettemann
Berlin School of Economics and Law, Germany, sebastian.huettemann@hwr-berlin.de

Roland M. Mueller
Berlin School of Economics and Law, Germany, roland.mueller@hwr-berlin.de

Kai R. Larsen
University of Colorado, Boulder, USA, kai.larsen@colorado.edu

Barbara Dinter
Chemnitz University of Technology, Chemnitz, Germany, barbara.dinter@wirtschaft.tu-chemnitz.de

Joshua Campos Chiny
Berlin School of Economics and Law, Germany, chiny.jc@gmail.com

How Best to Hunt a Mammoth - Toward Automated Knowledge Extraction from Graphical Research Models

Research Paper

Sebastian Huettemann[1], Roland M. Mueller[1], Kai R. Larsen[2], Barbara Dinter[3], and Joshua Campos Chiny[1]

[1] Berlin School of Economics and Law, Berlin, Germany
{sebastian.huettemann,roland.mueller}@hwr-berlin.de, chiny.jc@gmail.com
[2] University of Colorado, Boulder, USA
kai.larsen@colorado.edu
[3] Chemnitz University of Technology, Chemnitz, Germany
barbara.dinter@wirtschaft.tu-chemnitz.de

Abstract. In the Information Systems (IS) discipline, central contributions of research projects are often represented in graphical research models, clearly illustrating constructs and their relationships. Although thousands of such representations exist, methods for extracting this source of knowledge are still in an early stage. We present a method for (1) extracting graphical research models from articles, (2) generating synthetic training data for (3) performing object detection with a neural network, and (4) a graph reconstruction algorithm to (5) storing results into a designated research model format. We trained YOLOv7 on 20,000 generated diagrams and evaluated its performance on 100 manually reconstructed diagrams from the Senior Scholars' Basket. The results for extracting graphical research models show a F1-score of 0.82 for nodes, 0.72 for links, and an accuracy of 0.72 for labels, indicating the applicability for supporting the population of knowledge repositories contributing to knowledge synthesis.

Keywords: Knowledge Extraction, Graphical Research Models, Object Detection, Theory Repositories, Knowledge Synthesis

1 Introduction

Long before humans started to write letters, they painted pictures. Research suggests that some of the paintings in one of the most outstanding testaments of early human culture – the Lascaux Cave – depict hunting strategies that were used for education purposes and thereby knowledge sharing (Groeneveld, 2016; Maier et al., 2021). Such hunting strategies could be seen as early forms of theories depicting relationships between causes and effects: ten hunters, six spears, four axes, and setting a proper trap lead to a first surprised, then angry, and eventually dead mammoth.

Long after humans started to write letters, they still paint pictures. In a way, performing research in IS today is similar to hunting a mammoth ten thousand years

18th International Conference on Wirtschaftsinformatik,
September 2023, Paderborn, Germany

ago – with slight differences regarding the research focus. Instead of trying to identify theories about relationships between deadly instruments and hunting success, today's researchers try to identify theories about relationships between Information Technology and human behavior.

However, even after hundreds of thousands of years, we still share a central challenge with our ancestors: synthesizing our knowledge. For early humans, the only way to extend their knowledge might have been by exchanging information with hunters from different tribes, most likely by inspecting their neighbors' cave paintings. Fortunately, today's theories are no longer carved in stone but into paper and digital media. Unfortunately, the amount of available articles has grown to an immense body of knowledge that expands continuously and fast (Bornmann et al., 2021). This can lead to a range of unwanted side effects, such as information overload and knowledge fragmentation where conventional approaches to literature reviews tend to miss relevant articles (Larsen et al., 2019).

There have been attempts to create theory repositories in the form of databases to support researchers in synthesizing information from constructs, their definitions, and semantic relations (Mueller, 2015; Dann et al., 2019; Li et al., 2020). However, populating such databases is a major undertaking as automated methods for knowledge extraction are still in an early phase (Scharfenberger et al., 2021). Most existing approaches (Li & Larsen, 2011; Mueller & Huettemann, 2018; Mueller & Abdullaev, 2019) use natural language processing (NLP) for exclusively analyzing the text of papers. In addition to text, illustrating and reporting research findings in the form of diagrams evolved into well-accepted practice and is at the core of many successful research articles. In the IS discipline, graphical research models are widely used, providing an overview of factors or theoretical constructs in empirical settings, experiments, and surveys (Palvia et al., 2006; Recker, 2013; Kiessling et al., 2020).

Technical progress in the development of object detection algorithms based on deep learning technologies made it possible to extract information from graphical research models. A stream of research contributes to the population and maintenance of theory repositories in IS by aiming to automatically detect constructs, path coefficients, and items from research diagrams and structural equation models (Auer et al., 2013; Scharfenberger et al., 2021; Schoelch et al., 2022).

Some of the current limitations however include low variation in training data which might impede generalizability and challenges in inferring the relationships between individual elements in diagrams. We aim to contribute towards developing methods for knowledge extraction from graphical research models, and ask the following research question: *How can we improve the extraction of knowledge from graphical research models in scientific articles to support the population of theory repositories?*

In this article, we propose a method for (1) classifying and extracting graphical research models from scientific articles, (2) creating synthetic training data, (3) performing object classification, (4) re-constructing detected graph structures, and (5) persisting findings into a meta-model. We focus on the analysis of graphical research models (Palvia et al., 2006; Recker, 2013; Kiessling et al., 2020) whereas other types of diagrams, e.g., process models, reference frameworks, or organigrammes, have their own characteristics where for instance the shape of objects indicates special meaning. The proposed method could inform future work in other areas but would need to be modified towards interpreting the specific semantics of other types of diagrams.

2 Theoretical Background

2.1 Knowledge Extraction and Analysis from Text

Text mining and other NLP techniques have been the predominant means to extract and analyze knowledge from scientific articles. Li & Larsen (2011) presented a system for extracting constructs from IS papers. Larsen and Bong (2016) proposed a method for construct detection that surpassed latent semantic analysis. Li et al. (2020) presented a framework for extracting constructs and relationships from text. Mueller and Huettemann (2018) presented the tool CauseMiner to extract constructs and their interrelations from hypotheses and propositions in research articles. Mueller and Abdullaev (2019) presented DeepCause which extends CauseMiner by using a deep learning architecture for causal extraction.

However, these approaches share the challenge of locating, identifying, and extracting relevant information bits in the form of constructs and their interrelations from full-text scientific articles which can lead to ambiguous results.

2.2 Knowledge Extraction and Analysis from Diagrams

A number of studies aimed to extract figures from research articles. Clark and Divvala (2016) presented the algorithm PDFFigures 2.0 for extracting figures and tables. Their algorithm locates figures by reasoning about empty regions in text and identifying captions. Siegel et al. (2018) developed a deep neural model for detecting figures in PDF documents called DeepFigures. They approached the task of image classification and object detection by applying an OverFeat detection architecture to image embeddings. Their model detects bounding boxes for figures in PDF documents and was deployed in the academic search engine Semantic Scholar to extract figures from articles. Genz and Funk (2020) used convolutional neural networks (CNN) for the detection of structural equation models by converting pages of scientific papers to images.

Auer et al. (2013) presented an optical graph recognition approach attempting to perform the task of interpreting extracted figures. Their method was based on traditional image processing techniques, recognizing edges and their attachments to vertices. Attempts to detect objects and their interrelations in offline hand-drawn diagrams (Schaefer & Stuckenschmidt, 2019; Fang et al., 2022) utilized more sophisticated technologies, such as Recurrent-CNNs.

Closely related to this article, Scharffenerger et al. (2021) focussed on identifying and extracting knowledge from structural equation models in research articles. They identified and extracted figures from PDF articles via Recurrent-CNNs to detect constructs, path coefficients, and items via YOLOv4 (Bochkovskiy et al., 2020), a neural network for object detection. They used a rather small training dataset with 534 images and achieved good results.

Schoelch et al. (2022) developed an approach that included the automated re-creation of graphs, preserving information about the interrelations between objects. They created a synthetic dataset consisting of 12,000 diagrams to train YOLOv5 (Jocher et al., 2022). They performed object detection, text recognition per EasyOCR

(EasyOCR, 2023), and graph reconstruction by applying a set of heuristics to analyze the predicted bounding box information. For evaluating the performance on real diagrams, they manually labeled 24 diagrams from DISKNET (Dann et al., 2019). Their results indicate an accurate detection of nodes, but a less accurate detection of edges with a direction.

Previous approaches mentioned several limitations that might be addressed by future work. Scharffenberger et al. (2021) neither mapped text to bounding boxes nor path coefficients to edges. We also found that they omitted the detection of edges. Schoelch et al. (2022) mentioned problems in the detection and analysis of edges as well. In addition, they were not able to detect arrowheads and reported limitations in generating synthetic training data, such as low variation in nodes, arrow- and edge-types, and layouts that are often more randomly organized than human-structured diagrams. These aspects might decrease the generalizability of models trained on such input data.

With our approach, we aim to build on these limitations and suggestions for future work. We explore the usage of Graphviz to add more variety and structure to synthetic training data, and show that current versions of object detection algorithms are able to detect different types of nodes, arrows, and edges in synthetic as well as real-world diagrams. We further illustrate how to reconstruct a detected graph from bounding box information to infer relationships between nodes, and we compare and incorporate OCR technology to evaluate the extraction of text from nodes and edges.

3 Knowledge Extraction from Graphical Research Models

We aim to build on related work, applying state-of-the-art technologies to cover a complete workflow: from a corpus of research articles with diagrams to the extraction of constructs and their interrelations in a knowledge base. More than a case of incremental improvements in each task, this is about evaluating the potential of a whole pipeline of challenges. In such a setup, every error propagates through the pipeline rendering the final result as interpretable and useful, or not. Our proposed method consists of five steps. The complete flow from research articles to reconstructed diagrams is illustrated in Figure 1. We describe each step in the following subsections.

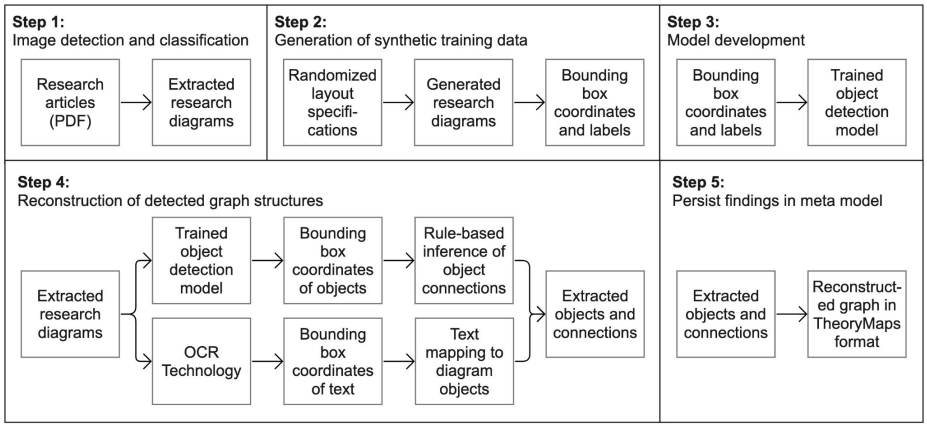

Figure 1. Knowledge extraction from graphical research models

3.1 Step 1: Image Detection and Classification from Research Articles

We analyzed 700 research articles published in the AIS Basket of eight (AIS, 2021) in 2018 and 2019 and extracted all figures from these articles by using the tool PDFFigures 2 (Clark & Divvala, 2016) resulting in a total of 922 images. We manually classified the images into two categories; those showing either graphical research models or the empirical results of a research model vs. all remaining figures.

We compared the performance of Naive Bayes, SVM, Random Forest, BERT, and SciBert by training them with only the captions and a combination of captions plus the text that was extracted from the figures themselves. We also implemented an Xception-model that was pre-trained on ImageNet and fine-tuned with the annotated images. In addition, we trained an ensemble classifier with the best-performing Naive Bayes, SVM, Random Forest, SciBERT, and Xception models. Eventually, we used the probabilities of the best-performing models to train a stacking classifier meta-model that outperformed all previous models. This classifier yielded an accuracy of 0.94 and a macro F1-score of 0.92.

All models that used text data performed better when the image text was also included in the dataset. We first did some general preprocessing on the data, including removing numbers, punctuation, multiple whitespaces, stop words, and short words, converting the text to lowercase, and applying lemmatization. We have also created a dummy classifier as a baseline which predicted the most frequent label for all the data. Table 1 shows detailed metrics of the top 5 best-performing models by macro F1-score.

Table 1. Image classification results of top 5 models by macro F1-scores

Model	Data	Precision	Recall	Macro-F1
Stacking Classifier	Fig + Cap + Text	0.93	0.91	0.92
SciBERT	Cap + Text	0.90	0.88	0.89
Ensemble Classifier	Fig + Cap + Text	0.92	0.85	0.88
SVM	Cap + Text	0.90	0.85	0.87
Xception	Fig	0.86	0.86	0.86
Dummy	N.A.	0.37	0.50	0.43

For the evaluation, we used standard measures in NLP where precision is calculated as true positives / (true positives + false positives), recall is calculated as true positives / (true positives + false negatives), and F1-score is calculated as 2 * (precision * recall) / (precision + recall). Macro F1-scores provide an unweighted average of F1-scores for the individual classes (SciKit Learn, 2023b; SciKit Learn, 2023a).

3.2 Step 2: Generation of Synthetic Training Data

For generating diagrams including their bounding boxes, we used Graphviz (Ellson et al., 2004; Graphviz, 2022) and developed a Python API. Our goal was to generate diagrams as training data that resemble graphical research models in research articles and contain as many variations in layouts and objects as possible.

The structure of the generated diagrams is based on a three-level hierarchical design. In a top-to-bottom diagram this would result in three levels: top, center, and bottom whereas, in a left-to-right diagram, the three levels are left, center, and right,

respectively. Each level can contain a random number of nodes and the distance between nodes and levels can be randomized as well. Connections between nodes are randomly defined. Nodes can only be connected to the next level, i.e., there are no connections between nodes from level one to level three.

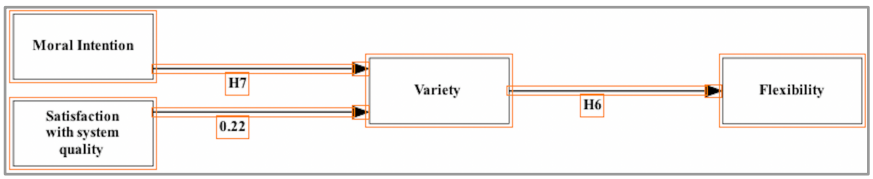

Figure 2. Generated training data example with bounding boxes

We defined four classes for training: nodes, edges, arrowheads, and edge labels. Figure 2 shows an example of a simple generated diagram where the respective bounding boxes for training are marked in orange. We defined bounding boxes for all classes as rectangles. We further applied extensive randomization across generated diagrams. To illustrate the variety in synthetic training data, Table 2 contains selected examples of randomization options.

Table 2. Diagram randomization examples

Attribute	**Randomization Options**
Node-shape	rectangle, ellipse, circle, only_text, rectangle_rounded_corners
Fonts	Arial, Helvetica, Times, Courier (normal, bold, italic)
Fontsize	8, 10, 12
Node-width	In inches: 1.3, 1.6, 2, 2.5
Node-height	In inches: 0.6, 0.8, 1, 1.3, 0.2
Graph-style	right-left, left-right, top-bottom, bottom-top
Edge-type	straight, curved, orthogonal
Edge-style	solid, dashed, dotted, bold

3.3 Step 3: Model Development

YOLO stands for "You Only Look Once" and denotes a series of object detection algorithms. Previous work used YOLOv4 (Scharfenberger et al., 2021) and YOLOv5 (Schoelch et al., 2022). In YOLOv6, improvements for detecting small objects were added (Olorunshola et al., 2023). YOLOv7 contained additional improvements regarding speed and accuracy and demonstrated a 13.7% increase in average precision compared to YOLOv6 (Wang et al., 2022).

We experimented with a pre-trained version of YOLOv7 (Wang et al., 2022) and performed fine-tuning by training the model with training sets of different sizes: 10,000, 20,000, 50,000, and 100,000. For each training set, we split the data into train, validation, and test sets with a ratio of 70/20/10. We did not find noticeable improvements in precision, recall, and mean average precision values when training a model with more than 20,000 images. One reason could be that the model internalized most of the possible layout structures given the selected randomization options.

We followed the guidelines for model development from the YOLOv5 documentation (Ultralytics, 2022) and additionally included 1,400 background images

without any labels to the training and 300 background images to the validation set. Setting the confidence threshold to 0.5 provided the best results.

Figure 3 shows selected evaluation results from the final model, trained on 20,000 synthetically generated diagrams. Training the model for 300 epochs, resulted in a precision of 0.99, a recall of 0.98, a mean average precision for a confidence threshold of 0.5 (mAP@0.5) of 0.99, and a mean average precision averaged over different confidence thresholds from 0.5 to 0.9 (mAP@0.5:0.95) of 0.86.

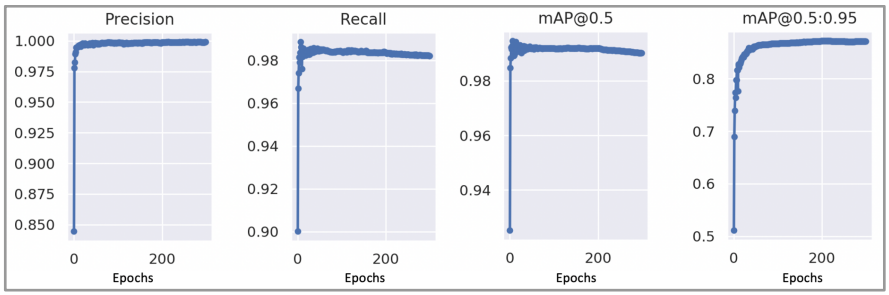

Figure 3. Selected metrics from YOLOv7 results

3.4 Step 4: Reconstruction of Detected Graph Structures

We created an algorithm to reconstruct the interconnections between nodes from the detected bounding box coordinates. Figure 4 illustrates an example of a simple extracted diagram. Each element is detected as a bounding box including a designated label, e.g., *node*, *arrowhead*, *edge,* or *edge label*. We iterate through all edge objects and check for overlaps with nodes and arrowheads. Whenever we find an arrowhead, we check for a connection with another node. We analyze the position of an arrowhead on an edge to get the direction of the arrow. For detecting labels, we take the label that is closest to the center of an edge. If an edge is diagonal, we can use the position of the arrowhead to infer the position of the other end of the edge.

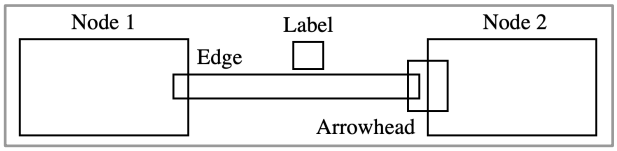

Figure 4. Detected bounding boxes and classes

We performed several tests with optical character recognition technologies, such as EasyOCR (EasyOCR, 2023) and Tesseract (Tesseract, 2023). EasyOCR provided better performance than Tesseract but was still not able to capture all texts correctly. The detection of edge labels, such as "H1", "0,39", and "+", worked only in a small fraction of cases. We therefore decided to use PaddleOCR (Du et al., 2020) which outperformed the other technologies. PaddleOCR outputs bounding box information of detected texts from images. We mapped the detected texts from PaddleOCR with the positions of nodes, edges, and edge labels in the reconstructed diagram to infer semantic connections between constructs.

3.5 Step 5: Persist Findings in Meta-Model

We propose a meta-model for graphical research models building up on the meta-model of Mueller for analyzing inter-theory relationships (Mueller, 2015). Figure 5 illustrates the meta model and its components.

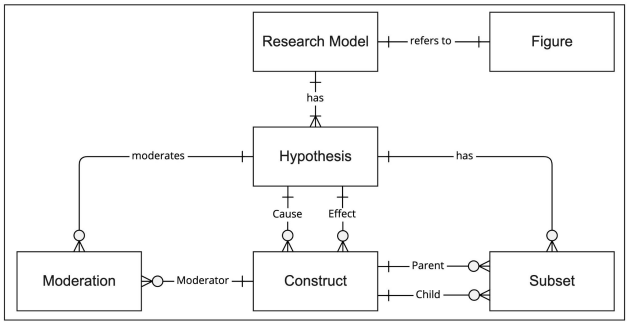

Figure 5. A meta-model for graphical research models

Diagrams can be coded into a research model. Such a structure is able to capture objects and their interrelations from research diagrams. In our implementation, we assume that every node in a diagram contains a construct. A construct can alternatively act as a moderator or be part of a subset where multiple constructs are grouped. The extraction of moderating relationships and grouped nodes is however not yet supported. We interpret two connected nodes in a diagram as two connected constructs where their connection implies a causal direction. Such connections between constructs form a hypothesis where the source construct represents a cause and the target construct an effect. Based on coding research diagrams according to our meta-model, we store extracted information from diagrams in a structured YAML format allowing for querying information for subsequent analysis steps.

4 Evaluation

4.1 Evaluation on Synthetic Diagrams

Our goal is to find out how well the proposed method is able to capture semantic relationships from research diagrams. We therefore generated 1,000 diagrams with randomized layouts as illustrated in Table 2. For each diagram, we generated a YAML file containing a research model description that represents constructs and their interrelations serving as ground truth. We performed object detection on the generated diagrams and used our algorithm for graph reconstruction to infer relationships between detected constructs. This resulted in a second YAML file containing the detection result for each diagram.

We developed a routine to compare the ground truth against the detection results and adjusted common machine learning metrics (precision, recall, F1) to evaluate the graph reconstruction task. We define precision as the fraction of correctly extracted objects (which can be nodes or links) among all extracted objects. We define recall as the fraction of correctly extracted objects among all true objects. A link is correct if it

connects the right two nodes with the correct direction. A node is only correctly identified, if the node text is correctly extracted. By allowing for a Levenshtein distance of three, we compensated for minor spelling mistakes due to OCR limitations where for instance whitespace is not detected correctly, e.g., *perceived easeof use*. We compared the number of nodes in each diagram by checking if a node with the same text was correctly detected. We also compared if the links between nodes were correctly set and whether related labels were identified. For correctly identified links, the label accuracy is 0.52. Table 3 shows the results of the evaluation for synthetic diagrams.

Table 3. Evaluation on synthetic diagrams

Metric	Nodes	Links
Absolute count in ground truth	8,949	7,969
Absolute count in detection results	8,949	7,832
Precision	0.99	0.98
Recall	0.99	0.96
F1	0.99	0.97

4.2 Evaluation on Real Diagrams

We extracted 573 graphical research models by performing image detection and classification (step 1) on research articles of the AIS Basket (AIS, 2021). We manually classified the extracted models into suitable (n=384) and not suitable (n=189) for automated extraction. Not suitable models utilized layouts that contained elements such as crossing edges that we did not include in our prototype. Although nodes in such models get detected correctly, the detection of correct links between nodes is not yet supported. From the 384 suitable models, we sampled 100 diagrams for manual evaluation. Figure 6 shows examples of supported and not supported layouts.

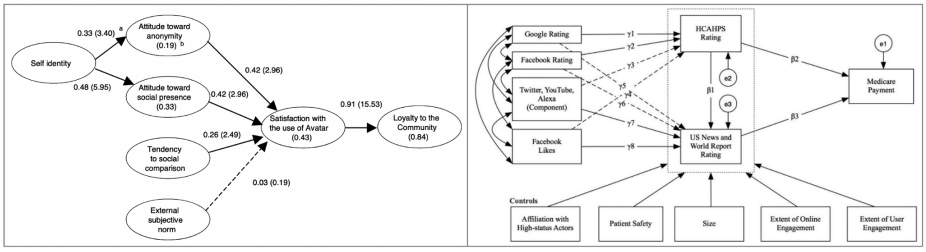

Figure 6. Examples for supported (left) and not supported layouts (right).
Left: Kim and Song (2004); right: Ivanov and Sharman (2018)

One of the authors and a master student in IS each translated 45 of the sampled diagrams into the proposed meta-model format (step 5) by manually creating YAML files. For calculating interrater reliability, both annotators annotated 10 diagrams and compared the results. We used the F1 score for measuring interrater reliability, due to the structure of the annotation task that does not allow us to use standard inter-annotator measures like Cohen's Kappa. The F1 score with a Levenshtein distance of three for nodes, links, moderations, and subsets, was 0.97, 0.92, 1.0, and 1.0, respectively. The resulting 100 manually recreated research diagrams form the gold standard.

We performed automated object detection and graph reconstruction on the extracted 100 diagrams resulting in the respective YAML files containing research models according to our meta-model. To compare the gold standard against the automatically detected research models, we used the same routine as for the evaluation of generated diagrams, including allowing for a Levenshtein distance of three to account for OCR limitations. The average accuracy of detected link labels is 0.72. Table 4 shows the results.

Table 4. Evaluation against manually annotated real diagrams

Metric	**Nodes**	**Links**
Absolute count in ground truth	664	581
Absolute count in detection results	583	526
Precision	0.88	0.76
Recall	0.8	0.7
F1	0.82	0.72

5 Discussion

We proposed and evaluated a method for extracting graphical research models from scientific articles in IS. By using state-of-the-art technologies for object detection and OCR, we described how knowledge depicted in research model diagrams can be extracted and stored in a format that supports the organization and discovery of constructs and their relationships to support populating knowledge repositories (Dann et al., 2019; Li et al., 2020) for the IS community.

With this approach, we have built up on the related works of Scharfenberger et al. (2021) and Schoelch et al. (2022) who developed similar approaches. In addition to the work of Scharfenberger et al. (2021), our method detects links between constructs and includes a graph reconstruction algorithm for the interpretation of detected bounding boxes. Thereby, we can associate constructs with each other and labels to edges enabling inference of causal directions. We further implemented OCR technology for text extraction to persist findings according to the proposed meta-model. In contrast to Schoelch et al. (2022), we integrated an object detection approach into a complete workflow proving its applicability in a scientific domain. We also advanced their approach by detecting the positions of arrowheads and training a neural network for object detection with a higher variety in node- and edge design. We further tested and implemented current technologies and introduced an algorithm for graph reconstruction, demonstrating that this approach works in a real-world scenario.

The results from the evaluation against 1,000 synthetic diagrams indicate a very accurate performance in detecting constructs (F1=0.99) and links (F1=0.97) with limitations in assigning the correct labels (accuracy=0.52) to links between nodes. The results from the evaluation against 100 manually reconstructed diagrams from scientific articles in IS show slightly lower results for the detection of constructs (F1=0.82), links (F1=0.72), and label assignments (accuracy=0.72).

We reviewed the annotated images of the model to identify potential reasons for the differences between manual and synthetic evaluation results. Some of the differences most likely resulted from lower image resolution of real-world diagrams which led to

incorrect OCR, bigger font size for labels which led to an increased accuracy score, and some not yet supported features such as grouped nodes and the detection of moderating relationships. We initially trained the model with diagrams that only included connections to nodes on adjacent levels as described in section 3.2. We found that the model was capable of generalizing well to the real-world dataset as it correctly identified edges connecting nodes on various levels.

5.1 Limitations

There are different sources of errors relating to object detection accuracy, OCR accuracy, and graph reconstruction accuracy. We found that horizontal lines pose a challenge to our trained object detection model. Missing such objects leads to follow-up errors as a connection between objects can not be inferred during graph reconstruction, resulting in missing links between constructs in our database. In addition, the detection of single-character edge labels, such as "+" or "-" poses a challenge for OCR. Both issues indicate problems in detecting objects with a low pixel density in very thin shapes. More extensive image preprocessing might help mitigate this shortcoming.

Our algorithm for graph reconstruction relies on interpreting labels and coordinates of bounding boxes. The main challenge is to infer the path of edges as diagonal edges are represented as rectangles. For our implementation, we based our inference on the pragmatic decision that every edge is interpreted as a straight edge. Based on this assumption, we can infer the path of an edge by analyzing the position of a connected arrowhead and thereby assign source and target nodes of a connection. Figure 7 illustrates this approach and also shows an example where a cornered edge is still interpreted correctly.

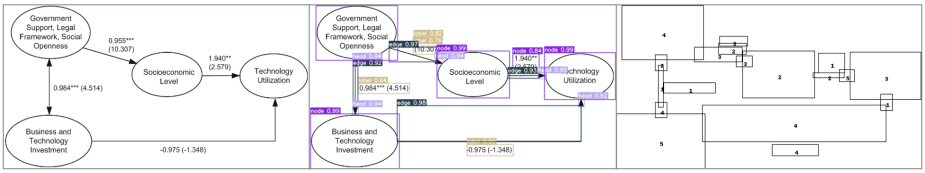

Figure 7. From real diagram to object detection results to bounding box inference

Although the approach works well in the majority of cases, it is not able to accurately detect paths of curved or cornered edges. This also impedes the accurate allocation of labels to edges. Using instance segmentation where instead of a bounding box the precise shape of an object is used for training a model could increase accuracy. However, it would also lead to additional complexity as these shapes contain much more detailed coordinates, leading to challenges in creating training data and in automatically reconstructing graphs to detect connections between objects.

5.2 Future Work

In order to accurately support the population of theory repositories allowing for analyzing constructs and their interrelations on a large scale, some challenges must be addressed to further improve the performance of this method. Our approach could be

enhanced by extending the graph reconstruction algorithm to add detection capabilities for moderating relationships and grouped nodes. However, we think that the hardest challenge is the correct detection of edges in diagrams. Using instance segmentation to get the exact shape of an arrow might be a way towards more overall accuracy. Such an approach would also enable the correct assignment of labels to edges, but it comes at the expense of higher effort in generating training data and fine-tuning a graph reconstruction algorithm for interpreting more complicated bounding box shapes.

6 Conclusion

In this article we presented a method for automated knowledge extraction from graphical research models covering a workflow from research articles to persisting findings in a designated data format. We (1) classified and extracted research diagrams from scientific articles, (2) created synthetic training data of high variance, (3) performed object classification with state-of-the-art technologies, (4) reconstructed detected graph structures and (5) stored our findings into a meta-model for graphical research models. The performance in correctly extracting, interpreting, and reconstructing real-world diagrams shows good results for the detection and interpretation of constructs (F1=0.82), links (F1=0.72), and labels (0.72). Future developments could incorporate instance segmentation techniques to overcome limitations in edge interpretation and label assignment.

As an introductory note, we boldly claimed that performing research in IS today - where researchers need to manually analyze hundreds of articles to get an overview of the state of the art - is not too different from the prehistoric approaches of our early ancestors to research better hunting strategies by comparing and assessing cave paintings across their friendly neighbors' caves.

Naturally, we do not know if our ancestors eventually managed to find the best theory to hunt a mammoth but we presented a method that can support researchers today in collecting data from thousands of articles reducing the amount of necessary manual labor. We hope that in the not-too-distant future, the work from months and sometimes even years can be reduced to minutes enabling new kinds of analyses tapping into the collective knowledge of researchers in IS. Similar to cave paintings that informed ancient hunters, the semi-automated extraction and synthesis of research diagrams could support researchers in setting out to hunt their next big scientific discovery.

References

AIS (2021), Senior Scholars' Basket of Journals, https://aisnet.org/page/SeniorScholarBasket, Accessed: 15.11.2021.

Auer, C., Bachmaier, C., Brandenburg, F.J., Gleißner, A. & Reislhuber, J. (2013), 'Optical Graph Recognition', *Journal of Graph Algorithms and Applications* **17**(4), pp. 541–565.

Bochkovskiy, A., Wang, C.-Y. & Liao, H.-Y.M. (2020), YOLOv4: Optimal Speed and Accuracy of Object Detection, http://arxiv.org/abs/2004.10934, Accessed: 7.11.2022.

Bornmann, L., Haunschild, R. & Mutz, R. (2021), 'Growth rates of modern science: a latent piecewise growth curve approach to model publication numbers from established and new literature databases', *Humanities and Social Sciences Communications* **8**(1), pp. 1–15.

Clark, C. & Divvala, S. (2016), PDFFigures 2.0: Mining Figures from Research Papers, *in* 'Proceedings of the 16th ACM/IEEE-CS on Joint Conference on Digital Libraries' ACM, pp. 143–152.
Dann, D., Maedche, A., Teubner, T., Mueller, B. & Meske, C. (2019), DISKNET – A Platform for the Systematic Accumulation of Knowledge in IS Research, *in* 'Proceedings of the 40th International Conference on Information Systems (ICIS)', p. 11.
Du, Y., Li, C., Guo, R., Yin, X., Liu, W., Zhou, J., Bai, Y., Yu, Z., Yang, Y., Dang, Q. & Wang, H. (2020), PP-OCR: A Practical Ultra Lightweight OCR System, http://arxiv.org/abs/2009.09941, Accessed: 10.11.2022.
EasyOCR (2023), EasyOCR - Github Repository, https://github.com/JaidedAI/EasyOCR, Accessed: 9.3.2023.
Ellson, J., Gansner, E.R., Koutsofios, E., North, S.C. & Woodhull, G. (2004), Graphviz and Dynagraph — Static and Dynamic Graph Drawing Tools, *in* Jünger, M. & Mutzel, P. (eds.) 'Graph Drawing Software' Mathematics and Visualization, Springer Berlin Heidelberg, pp. 127–148.
Fang, J., Feng, Z. & Cai, B. (2022), 'DrawnNet: Offline Hand-Drawn Diagram Recognition Based on Keypoint Prediction of Aggregating Geometric Characteristics', *Entropy* **24**(3), p. 425.
Genz, T. & Funk, B. (2020), Using CNNs to Detect Graphical Representations of Structural Equation Models in IS Papers, *in* 'WI2020 Zentrale Tracks' GITO Verlag, pp. 115–120.
Graphviz (2022), Graphviz - An Open Source Graph Visualization Software, https://graphviz.org/, Accessed: 7.11.2022.
Groeneveld, E. (2016), World History Encyclopedia - Lascaux Cave, https://www.worldhistory.org/Lascaux_Cave/, Accessed: 22.9.2022.
Ivanov, A. & Sharman, R. (2018), 'Impact of User-Generated Internet Content on Hospital Reputational Dynamics', *Journal of Management Information Systems* **35**(4), pp. 1277–1300.
Jocher, G. et al. (2022), YOLOv5 - Relase Version 6.2, https://zenodo.org/record/7002879, Accessed: 7.11.2022.
Kiessling, S., Figl, K. & Miniukovich, A. (2020), Graphical Research Models in the Information Systems Discipline, *in* 'Proceedings of the 53rd Hawaii International Conference on System Sciences'.
Kim, Y.J. & Song, J. (2004), Unveiling User Characteristics in Virtual Communities and the Impact on E-Commerce, *in* 'Proceedings of the International Conference on Information Systems (ICIS)', p. 15.
Larsen, K.R. & Bong, C.H. (2016), 'A tool for addressing construct identity in literature reviews and meta-analyses', *MIS Quarterly* **40**(3), pp. 1–23.
Larsen, K.R., Hovorka, D.S., Dennis, A.R. & West, J.D. (2019), 'Understanding the Elephant: The Discourse Approach to Boundary Identification and Corpus Construction for Theory Review Articles', *Journal of the Association for Information Systems* , pp. 887–927.
Li, J., Larsen, K. & Abbasi, A. (2020), 'TheoryOn: A Design Framework and System for Unlocking Behavioral Knowledge Through Ontology Learning', *MIS Quarterly* **44**(4), pp. 1733–1772.
Li, J. & Larsen, K.R. (2011), Establishing Nomological Networks for Behavioral Science: a Natural Language Processing Based Approach, *in* 'Proceedings of the International Conference on Information Systems (ICIS)'.
Maier, G.J., Musholt, E.A. & Stava, L.J. (2021), 'Lascaux Cave, Part Four: Evidence of Hunting', *Journal of Transpersonal Psychology* **53**(1).
Mueller, R. & Abdullaev, S. (2019), DeepCause: Hypothesis Extraction from Information Systems Papers with Deep Learning for Theory Ontology Learning, *in* 'Hawaii International Conference on System Sciences (HICSS)'.
Mueller, R.M. (2015), A Meta-Model for Inferring Inter-Theory Relationships of Causal

Theories, *in* '48th Hawaii International Conference on System Science (HICSS)' IEEE, pp. 4908–4917.
Mueller, R.M. & Huettemann, S. (2018), Extracting Causal Claims from Information Systems Papers with Natural Language Processing for Theory Ontology Learning, *in* 'Proceedings of the 51st Hawaii International Conference on System Sciences'.
Olorunshola, O.E., Irhebhude, M.E. & Evwiekpaefe, A.E. (2023), 'A Comparative Study of YOLOv5 and YOLOv7 Object Detection Algorithms', *Journal of Computing and Social Informatics* **2**(1), pp. 1–12.
Palvia, P., Midha, V. & Pinjani, P. (2006), 'Research Models in Information Systems', *Communications of the Association for Information Systems* **17**.
Recker, J. (2013), *Scientific Research in Information Systems*. Springer.
Schaefer, B. & Stuckenschmidt, H. (2019), Arrow R-CNN for Flowchart Recognition, *in* '2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)' IEEE, pp. 7–13.
Scharfenberger, J., Funk, B. & Mueller, B. (2021), The Augmented Theorist - Toward Automated Knowledge Extraction from Conceptual Models, *in* 'International Conference on Information Systems (ICIS)'.
Schoelch, L., Steinhauser, J., Beichter, M., Seibold, C., Yang, K., Knäble, M., Schwarz, T., Mädche, A. & Stiefelhagen, R. (2022), Towards Automatic Parsing of Structured Visual Content through the Use of Synthetic Data, *in* '26th International Conference on Pattern Recognition (ICPR)'.
SciKit Learn (2023a), Example of Precision-Recall metric to evaluate classifier output quality, https://scikit-learn/stable/auto_examples/model_selection/plot_precision_recall.html, Accessed: 10.3.2023.
SciKit Learn (2023b), SciKit Learn Metrics - sklearn.metrics.f1_score, https://scikit-learn/stable/modules/generated/sklearn.metrics.f1_score.html, Accessed: 5.3.2023.
Siegel, N., Lourie, N., Power, R. & Ammar, W. (2018), Extracting Scientific Figures with Distantly Supervised Neural Networks, *in* 'Proceedings of the 18th ACM/IEEE on Joint Conference on Digital Libraries', pp. 223–232.
Tesseract (2023), Tesseract OCR - Github Repository, https://github.com/tesseract-ocr/tesseract, Accessed: 9.3.2023.
Ultralytics (2022), YOLOv5 Documentation - Tips for Best Training Results, https://docs.ultralytics.com/tutorials/training-tips-best-results/, Accessed: 9.11.2022.
Wang, C.-Y., Bochkovskiy, A. & Liao, H.-Y.M. (2022), YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, http://arxiv.org/abs/2207.02696, Accessed: 7.11.2022.