Image Analysis of Structured Visual Content

Maximilian Beichter
Karlsruher Institute for Technology

Lukas Schölch
Karlsruher Institute for Technology

Jonas Steinhäuser
Karlsruher Institute for Technology

Abstract

Structured visual content such as graphs, flow charts or similar cannot be perceived by visually impaired people and machines. To make this possible, we propose an approach that converts such structured visual content in to a machine-understandable representation. Previous work exists only for very restricted subfields of this task which make strong assumptions about the images. To circumvent these restrictions, we propose an approach using machine learning, which tries to solve the problem for real data by using a synthetic data set. We showed, that this is successful to some degree and discuss the limitations of this approach and propose improvements for future applications.

1. Introduction

As concepts get more and more complex it gets harder to explain them with words. Therefore, many explanations, presentations and research papers use visually structured content such as flow charts and graphs to aid understanding. While this is a great possibility to condense a lot of information, images are not comprehensible for everyone. Visually impaired humans have so far no chance of gaining any additional information via graphical representations by themselves. Furthermore, knowledge extraction or searches in those are impossible, since machines can not comprehend visually structured content.

This paper is going to discuss old approaches to solve this task and its problems and presents a novel way to conquer this task with the help of machine learning. In that approach we also show the benefits of synthetic to real learning to mitigate the lack of ground truth annotations.

2. Related Work

So far little work has been done on recognition of structured visual content in general. Existent research focuses on various different subcategories of this field. Awal *et al*. [4] try to extract a graph like structure from hand drawn flow charts. This separates the task into two parts. Recognizing the elements that represent the graphical components and extracting the text itself. Auer *et al*. [3] on the other hand do not consider text and concentrate on recognizing limited graph structures. Those consist only of fully colored circular nodes and bidirectional straight edges. Their work uses a multi-step procedure with pre-processing, segmentation, topology recognition, and post-processing, while mainly using conventional computer vision methods and pixel based operations. Vasudevan *et al*. [9] go one step further, their goal is to directly extract the knowledge from flow charts without recognizing an explicit graph structure. Similarly, Wu *et al*. [10] want to generate code from given flow charts. Both of these works make strict assumptions about the given images and focus strictly on their use case.

In contrast, our approach aims at structured visual content more in general and does not demand a certain type of graph structure. Therefore, we want to detect basic elements such as nodes and their content, as well as different types of edges and their weights. The final graph representation should be purely descriptive, without being specialized for any certain use case.

3. Methods

Our approach mainly addresses the problem of mapping from an image showing a graph structure to a set of nodes and edges. In the following, we name this resulting set of nodes and edges graph representation. Therefore, our task describes the function

$$Image \rightarrow \{Node\} \cup \{Edge\}.$$

Nodes in our case are defined as their unique identifier and their containing text as content. Optional, a node can contain a set of sub-nodes to define aggregations of nodes called groupings. Similar to nodes, edges are also defined by their identifier and their weights. In addition to this an edge needs a start- and an end- node.

3.1. Traditional Approach: Computer Vision

Conventional computer vision methods use mostly pixel-based operations to gain structural information. The main task here is to differentiate between text and graph structures in the image. Therefore, we used different methods to extract coherent regions of certain size and shape. These methods depend largely on the given image and its quality. Different images follow different rules in their content and therefore need to be treated individually. Another challenge to solve are interrupted lines. These need to be detected and reconstructed accordingly.

Due to these challenges and various problems, that are discussed in the Discussion, we did not pursue this approach any further.

3.2. Proposed Approach: Synthetic to Real

In comparison to the conventional approach we use methods of machine learning, in particular we use object detection to recognize structures. In our case these are nodes and edges. As can be seen in Figure 1, an OCR is used in parallel to extract the texts with their bounding boxes from the image. Afterwards, the two results are combined to create a graph representation with the corresponding text assignment.

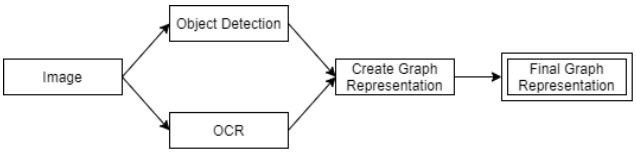

Figure 1. Showing the full Pipeline Scheme to generate the Graph Representation.

However, object detection is a supervised learning method, which means, we need labeled data to train a model. Since we do not have such data, we try to generate synthetic images with corresponding annotations. To achieve the best performances on real data, this synthetic data must be as realistic as possible.

3.2.1 Synthetic Data Generation

The goal is to generate images, that mimic a given set of real data while exhibiting high variation. Since an infinite amount of variation is not feasible, we have decided on the following properties: Nodes exist as rectangles or ellipses. A node can contain text and/or other nodes. Edges always run between two nodes as a direct or angled line. They can also have arrowheads at both, one or none of its ends. The line can be solid or dashed, and the appearance of the arrowhead can vary. The weights of an edge rest on it or intersect the line. Also, for all elements, the size and line thickness can vary.

In our implementation, nodes are distributed on a grid structure, varying in their position and shape. Their text content is selected from a list [2]. Then a number of edges is placed between nodes, no connection is taken twice, and care is taken that an edge does not collide with a node, except for connections that run into a grouping. Along such an edge, the weight, a number between 0 and 1, is written to a random position. Additionally, background text is added.

All of these actions are taken randomly, determined by a set of hyper-parameters. Therefore, we have generated a more difficult data set than a real data set, as it contains various anomalies. This is due to the generous use of random ranges, which sometimes creates more complexity than is needed later. In total, we generated 12000 images, with a 70%, 20% and 10% split into train, validation and test data. Additionally, we generated all bounding box annotations, text bounding boxes and ground truth graph representations for possible later evaluations.

3.2.2 Object Detection

The objects generated in our data set are defined by a rectangular bounding box and a class label. Both nodes and groupings have the same class. Edges on the other hand are differentiated by their start and end points. Diagonal edges and angled edges go from one corner of their bounding box to another one (e.g. bottom-left to top-right). Horizontal and vertical edges go from the center of one side to the center of the opposite side of their bounding box (e.g. left to right). An edge is labeled horizontal or vertical if it is within +/- 2.5 degrees of being perfectly horizontal or vertical. Therefore, we have 4 diagonal classes, 2 horizontal and 2 vertical. Additionally, an edge could be bi-directional or non-directional, this leads to further 8 classes respectively.

The model used for the object detection is YOLOv5 [6] with the large weight configuration. Important for our class labels is that in the data augmentation step, no flipping is involved. Furthermore, the IoU threshold in training and detection should be set to zero, as overlapping bounding boxes are normal in this application.

3.2.3 OCR

Since text is an elementary component of graphs, it must also be included in the final graph representation. Because our object detection deals only with the detection of the structure, we used *easy-OCR* [1] for the text. This OCR works parallel to the object detection (see Figure 1) and has no dependencies to it. It takes the same image as the object detection and returns a list of all detected texts, as well as their bounding boxes and positions. These can be used later in the graph creation for the assignment.

3.2.4 Create Graph Representation

The results from object detection and OCR are now merged into a graph representation. This can be divided into two main steps:

Step 1 Construction of the graph structure:
In this process, groupings are detected at the beginning. To do this, the overlaps of all bounding boxes are determined. If a box is almost completely inside another, it is added to the larger node as a subnode. Afterwards, the nodes are connected with edges. For this, the *connection points* are searched for an edge on its bounding box. The connection points represent the start and end points of an edge. For a diagonal edge this would be a corner of a bounding box, for a horizontal edge, this would be the center of one bounding box side. For each connection point the next adjacent node is searched and connected to the edge. In the end, all elements are connected accordingly and we have created a graph structure, that contains all previously detected objects.

Step 2 Text mapping:
After the graph structure has been created, the texts have to be assigned to the corresponding elements in the graph. First, the overlap to the node boxes is determined for each text bounding box. If the overlap is large enough, the text is considered as a part of the node and added to it. The remaining texts must therefore belong to edges, or have been placed freely in the image. To assign the edges, all possible edge bounding boxes are determined for each text bounding box, based on their overlap. Then the most probable edge bounding box for the text is determined, based on all possible courses of an edge within its bounding box. All remaining texts are assumed to be background texts and will not be added to the graph representation.

4. Evaluation

This section focuses first on the data sets used for the evaluation, then on the applied metrics and finally on the most important numbers regarding the evaluation. For deeper insights please consider reading the Appendix. We will always distinguish between the object detection, the graph structure and the OCR with the text assignment.

4.1. Different Data Sets

To evaluate our pipeline we use four different data sets. We can divide these data sets into two groups, synthetic data and real data.

The synthetic data is generated exactly like the training data set and therefore has the same properties. From this data set we take a subset, which consists only of human understandable graphs, which we call adjusted synthetic test set in the following.

For the real world data set we manually annotated a random subset from DISKNET [5] images. As this data contains behavior, that is not covered in our synthetic data set we created an adjusted subset. This adjusted data set contains only behavior included in our synthetic data set generation.

4.2. Metrics

In the following, we present the metrics we used to evaluate our results. To evaluate the object detection we use common metrics. To evaluate the graph structure as well as the OCR and text assignments, we define our own metrics, since this is a problem without well established metrics.

4.2.1 Metrics for the Object Detection

To evaluate the object detection we use the mAP defined as

$$mAP = \frac{1}{|classes|} \sum_{c \in classes} \frac{\#TP(c)}{\#TP(c) + \#FP(c)},$$

with a true positive (TP) as a prediction bounding box A with an $IoU(A, B)$ to a groundtruth bounding box B,

$$IoU(A, B) = \frac{A \cap B}{A \cup B},$$

greater than a threshold, otherwise it is a false positive (FP). Furthermore, we evaluate the mAP averaged for IoU thresholds from 0.5 to 0.95 with a step size of 0.05 (COCO's [8] standard metric, simply denoted as $mAP@[.5, .95]$).

4.2.2 Metrics for the Graph Structure

Evaluating graphs and their isomorphism is a complex problem. After the prediction the results contain a graph representation with nodes and edges. But it contains no information about which structure of the predicted graph is related to which structure in the ground truth graph representation. As we get the bounding boxes of the predicted structures with our model we identify the relations between the predicted structures and the ground truth. We define

$$IsomorphicError = \#mNodes + \#mEdges,$$

with $mNodes$ and $mEdges$ as the amount of nodes and edges in the prediction and the ground truth representation which could not find a match. Therefore, this node or edge is considered as missing in the ground truth or in the prediction. According to this we also define the normalized version

$$NormIsomorphicError = \frac{IsomorphicError}{\#Nodes + \#Edges}$$

to be able to better compare graphs of different size. In addition, we also calculated this error measure only for the nodes as well as the edges. We take only the missing nodes and the missing edges for the isomorphic error and normalize them with the number of nodes or edges in the graph.

4.2.3 Metrics for the Text Mapping and the OCR

To compare the results of our text recognition as well as our assignment of the texts within the pipeline we define another error measure. We use the Levenshtein Distance [7]. The Levenshtein Distance $L(X, Y)$ is defined as the minimum number of deletions, insertions and replacement operations to change text X to text Y. With this we can define our

$$OCRError = \frac{\sum_{S_{matched}} L(GT_{content}, P_{content})}{\sum_{S_{matched}} \#GT_{content}},$$

with $S_{matched}$ as every matched structure, $GT_{content}$ as the content of the structure in ground truth, $P_{content}$ as the content of the structure in prediction and the amount of characters in the ground truth content $\#GT_{content}$.

4.3. Results

In the following sections we present the results, divided into the object detection part, the graph representation and the text recognition as well as their assignment.

4.3.1 Evaluation of the Object Detection

As we only obtain bounding box labels on the synthetic data we evaluate the object detection only on the synthetic test data set as visible in Table 1.

Class	Labels	mAP@.5:.95:
all	16619	0.676
node	9381	0.995
arrow_tl2br	1248	0.867
arrow_t2b	114	0.376
arrow_tr2bl	1294	0.878
arrow_r2l	254	0.486
arrow_br2tl	1289	0.873
arrow_b2t	110	0.476
arrow_bl2tr	1271	0.863
arrow_l2r	227	0.489
arrow_bi_tl2br	643	0.799
arrow_bi_t2b	58	0.436
arrow_bi_tr2bl	610	0.779
arrow_bi_r2l	120	0.47

Table 1. Showing the results of the object recognition on synthetic test set.

4.3.2 Evaluation of Graph Representations

Table 2 shows the results of the normalized Isomorphic Error with direction on the four different data sets. The model performed best on the adjusted synthetic data set. The mean Isomorphic Error of DISKNET test set does not differ a lot to the adjusted DISKNET test set.

Data Set	Mean	Std.
Synthetic Test Set	0.134	0.132
Adjusted Synthetic Test Set	0.043	0.057
Disknet Test Set	0.328	0.139
Adjusted Disknet Test Set	0.325	0.139

Table 2. Showing the Isomorphic Error results on the different data set.

If we look at the errors in detail, we see that the edges cause more errors than the nodes, regardless of the data set. An example of this is shown in Table 3, presenting the synthetic data set.

Structure	Mean	Std.
Nodes	0.016	0.0479
Edges	0.287	0.281

Table 3. Isomorphic Error Edges and Nodes on the Testset normalized.

4.3.3 Evaluation of Text Mapping and the OCR

In the evaluation of text recognition and text assignment, we observe the worst result in the synthetic test set. The best OCRError we observe in the adjusted DISKNET test set.

Data Set	Mean	Std.
Synthetic Test Set	0.423	0.429
Adjusted Synthetic Test Set	0.29	0.115
Disknet Test Set	0.254	0.214
Adjusted Disknet Test Set	0.20	0.187

Table 4. Showing the OCR Error results on the different data set.

5. Discussion

The following sections discuss the different shortcomings and possible improvements of our approach.

5.1. Traditional Approach: Computer Vision

This method is highly dependent on the given image and needs to be adjusted accordingly. Therefore, good results are only achievable with heavy fine-tuning. Since our task involved real world images with lots of variation, this approach is not feasible. The data varies too much to create a fixed set of rules that applies on all images.

5.2. Proposed Approach: Synthetic to Real

In our Synthetic to Real approach we can divide the limitations into two categories. One category deals with the limitations caused by the influence of the synthetic data set. The second category focuses on the limitations caused by the applied method.

5.2.1 Limitations: Data Set

The synthetic data set provides first good approaches to generate different graphs. However, there are many variations that can be added to this data set. For example, round edges and multiple nested groupings are missing. Also, more shapes of nodes would be a useful extension. But also more exotic representations, like combined edges are missing. Furthermore, the data set does not contain any additional information, such as legends. These would be an interesting special case, since they often show elements of the graph without having a semantic meaning for the graph itself. For further improvements it would be interesting to extend the synthetic data set to be able to represent even more graphs of the real world.

5.2.2 Limitations: Methods

One of our main methodological limitations is the heuristic assignment of text to edges. This constraint originates in the problem, that we only work with bounding boxes of edges and lost the information about their exact path. Additionally, real world images do not follow any rule on how to assign weights to edges and therefore it is difficult to design a general heuristic. Similarly, our nodes are detected and processed only by their bounding boxes too. Even though this did not lead to any problems now, it maybe would be better to switch from an object detection via bounding boxes to segmentation masks, especially for better edge detections.

6. Conclusion

Recognizing structured visual content is at least a difficult if not impossible task for visually impaired humans and machines. Our proposed method produced good results on this previously little explored topic. The goal of being able to process structured visual content without great assumptions about their composition was reached with some limitations. The overall capability of this approach is mainly dependent on the data used for the training. Here we generated a synthetic data set, following different rules. Even though this data set has its limitations (see Limitations) it was able to provide enough information to the model, to learn important aspects of structured visual content. This also showed, that the training data set can be harder than the real world data. Future work could extend this data set to include more components and variations. Furthermore, changing the detection step from bounding boxes to segmentation maps could lead to more improvements.

References

[1] Easyocr website. https://www.jaided.ai/easyocr/. Accessed: 2021-07-30. 2

[2] Massachusetts institute of technology: list of words. https://www.mit.edu/ ecprice/wordlist.10000. Accessed: 2021-07-30. 2

[3] Christopher Auer, Christian Bachmaier, Franz J. Brandenburg, Andreas Gleißner, and Josef Reislhuber. Optical graph recognition. In Walter Didimo and Maurizio Patrignani, editors, *Graph Drawing*, pages 529–540, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. 1

[4] Ahmad Montaser Awal, Guihuan Feng, Harold Mouchère, and Christian Viard-Gaudin. First experiments on a new online handwritten flowchart database. volume 7874, pages 1–10, 01 2011. 1

[5] David Dann, Alexander Maedche, Timm Teubner, Benjamin Mueller, Christian Meske, and Burkhardt Funk. Disknet – a platform for the systematic accumulation of knowledge in is research. 11 2019. 3

[6] Glenn Jocher, Alex Stoken, Jirka Borovec, NanoCode012, Ayush Chaurasia, TaoXie, Liu Changyu, Abhiram V, Laughing, tkianai, yxNONG, Adam Hogan, lorenzomammana, AlexWang1900, Jan Hajek, Laurentiu Diaconu, Marc, Yonghye Kwon, oleg, wanghaoyang0106, Yann Defretin, Aditya Lohia, ml5ah, Ben Milanko, Benjamin Fineran, Daniel Khromov, Ding Yiwei, Doug, Durgesh, and Francisco Ingham. ultralytics/yolov5: v5.0 - YOLOv5-P6 1280 models, AWS, Supervise.ly and YouTube integrations, Apr. 2021. 2

[7] V. I. Levenshtein. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. *Soviet Physics Doklady*, 10:707, Feb. 1966. 4

[8] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 3

[9] Bintu G. Vasudevan, Sorawish Dhanapanichkul, and Rajesh Balakrishnan. Flowchart knowledge extraction on image processing. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 4075–4082, 2008. 1

[10] Xiang-Hu Wu, Ming-Cheng Qu, Zhi-Qiang Liu, and Jian-Zhong Li. Research and application of code automatic generation algorithm based on structured flowchart. *JSEA*, 4:534–545, 01 2011. 1

A. Isomorphic Error on different Data Sets

A.1. Synthetic Test Set

A.1.1 Full Graph

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	1.930	2.025	[0,13]
✗	✗	1.170	1.699	[0,11]
✓	✓	0.134	0.132	[0,0.833]
✗	✓	0.081	0.115	[0,0.833]

A.1.2 Nodes

Norm.	Mean	Std.	[Min,Max]
✗	0.126	0.351	[0,2]
✓	0.016	0.0479	[0,0.5]

A.1.3 Edges

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	1.805	1.853	[0,12]
✗	✗	1.045	1.484	[0,9]
✓	✓	0.287	0.281	[0,3]
✗	✓	0.168	0.244	[0,3]

A.2. Adjusted Synthetic Test Set

A.2.1 Full Graph

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	0.645	0.863	[0,3]
✗	✗	0.419	0.752	[0,3]
✓	✓	0.043	0.057	[0.0,0.2]
✗	✓	0.03	0.053	[0.0,0.2]

A.2.2 Nodes

Norm.	Mean	Std.	[Min,Max]
✗	0.065	0.246	[0,1]
✓	0.009	0.035	[0.0,0.143]

A.2.3 Edges

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	0.581	0.872	[0,3]
✗	✗	0.355	0.743	[0,3]
✓	✓	0.1	0.163	[0.0,0.667]
✗	✓	0.068	0.152	[0.0,0.667]

A.3. Disknet Test Set

A.3.1 Full Graph

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	6.333	5.217	[1,22]
✗	✗	5.75	4.746	[1,21]
✓	✓	0.328	0.139	[0.091,0.654]
✗	✓	0.297	0.122	[0.091,0.556]

A.3.2 Nodes

Norm.	Mean	Std.	[Min,Max]
✗	0.542	1.322	[0,6]
✓	0.037	0.076	[0.0,0.273]

A.3.3 Edges

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	5.792	4.32	[1,16]
✗	✗	5.208	3.797	[1,15]
✓	✓	0.613	0.233	[0.143,1.0]
✗	✓	0.553	0.211	[0.143,1.0]

A.4. Adjusted Disknet Test Set

A.4.1 Full Graph

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	5.059	4.094	[1,17]
✗	✗	4.471	3.5	[1,13]
✓	✓	0.325	0.139	[0.091,0.654]
✗	✓	0.291	0.123	[0.091,0.556]

A.4.2 Nodes

Norm.	Mean	Std.	[Min,Max]
✗	0.235	0.73	[0,3]
✓	0.021	0.062	[0.0,0.25]

A.4.3 Edges

Dir.	Norm.	Mean	Std.	[Min,Max]
✓	✗	4.824	3.714	[1,16]
✗	✗	4.235	3.078	[1,12]
✓	✓	0.622	0.226	[0.2,1.0]
✗	✓	0.558	0.213	[0.2,1.0]

B. OCR Error

B.1. Synthetic Test Set

Mean	Std.	[Min,Max]
0.423	0.429	[0.0,13.0]

B.2. Adjusted Synthetic Test Set

Mean	Std.	[Min,Max]
0.29	0.115	[0.091,0.474]

B.3. Disknet Test Set

Mean	Std.	[Min,Max]
0.254	0.214	[0.039,0.692]

B.4. Adjusted Disknet Test Set

Mean	Std.	[Min,Max]
0.2	0.187	[0.039,0.692]