Aalborg Universitet

AALBORG UNIVERSITY
DENMARK

Generalizing Floor Plans using Graph Neural Networks

Simonsen, Christoffer Plovmand; Thiesson, Frederik Myrup; Philipsen, Mark Philip; Moeslund, Thomas B.

GENERALIZING FLOOR PLANS USING GRAPH NEURAL NETWORKS

Christoffer P. Simonsen   *Frederik M. Thiesson*   *Mark P. Philipsen*   *Thomas B. Moeslund*

Aalborg University
Visual Analysis and Perception Laboratory
Rendsburggade 14, 9000 Aalborg, Denmark

ABSTRACT

The proliferation of indoor maps is limited by the manual process of generalizing floor plans. Previous attempts at automating similar processes use rasterization for structure. With Graph Neural Networks (GNN) it is now possible to skip rasterization and rely on the inherent structures in CAD drawings. A core component in floor plan generalization is localization of doors. We show how floor plan graphs can be extracted directly from CAD primitives and how state-of-the-art GNNs can be used to classify graph nodes as door or non-door. Generalization is represented by the creation of placeholder bounding boxes using the labelled graph nodes. Our graph-based approach completely outperforms the Faster R-CNN baseline, which fail to locate any doors with the desired localization accuracy. To support further development of graph-based methods and comparison with raster-based methods, we publish a new dataset that consists of both image and graph-based floor plan representations. Code and dataset is available at `https://github.com/Chrps/MapGeneralization`.

Index Terms— graph neural networks, dataset, floor plan, indoor map

1. INTRODUCTION

Cartography has historically been a major driver of progress, enhancing our understanding of and ability to navigate the world. With the introduction of smartphones and readily available digitized maps the reliance on maps is now greater than ever. While outdoor maps have become ubiquitous and indispensable, indoor maps are far less commonplace. The global indoor positioning and navigation market is predicted to expand from \$2.6bn in 2017 to \$43bn by 2025 [1]. This can be helped by automating the creation of indoor maps. The impact of an automated and scalable process is difficult to anticipate considering the effect that accessible outdoor maps have had. For now, indoor maps are mostly known from larger venues, such as airports, stadiums, HQs, and universities, where people are likely to be unfamiliar with the layout.
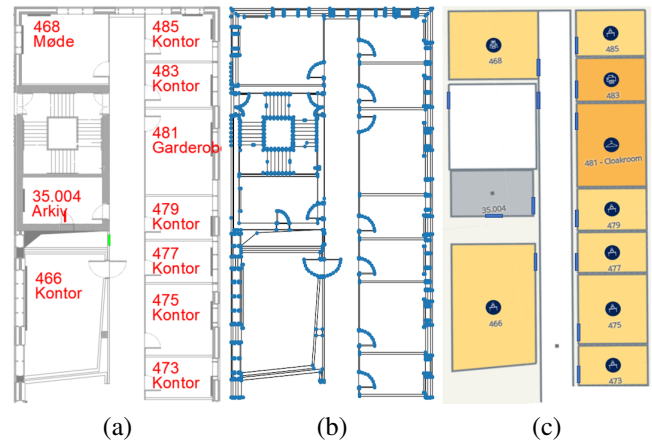
Fig. 1. (a) Rasterized floor plan. (b) Graph extracted from CAD primitives. (c) Generalized map for navigation.

Maps for indoor navigation must present information about room layout, hallways, and openings in a simple and standardized manner. In other words, a generalized representation. Specialists generalize maps with basis in existing technical drawings of buildings. These technical drawings are highly complex and can vary in terms of format, level of detail, and style. The time-consuming generalization process involves tracing the perimeter of rooms, locating doors, assigning labels, etc. An example of a typical floor plan image is shown in Figure 1 (a). This is the common representation used in related work as input to e.g. a Convolutional Neural Network (CNN). In this work we investigate an alternative representation based on graphs. Such a graph, with blue nodes and gray edges, is shown in Figure 1 (b). The graph is created by parsing a vector-based CAD file. This work is a step towards automatically generating generalized maps for indoor navigation similar to the one shown in Figure 1 (c). Here we focus on the central problem of localizing doors.

Automatic analysis of technical drawings in general and floor plans in particular has long been an active research area, with methods overwhelmingly operating in the image domain (like Figure 1 (a)). Most recent methods rely on Deep Neural

Networks (DNN) for detecting rooms [2, 3] and doors [4, 2, 3] and segmenting walls [5, 3]. Prior to the Deep Learning revolution, methods were based on simple rules and classic image processing techniques such as Hough transform for detecting walls [6] and morphological operations for segmenting text [7] or rooms [8]. Common for most image based approaches is the amount of effort going into recovering information that has been obfuscated or lost because of rasterization. One example is Raster-to-Vector [9], where vectorized representations of floor plans are created by extracting entities such as wall junctions using a CNN. The information loss and inefficiencies from rasteriztion are the result of quantisation and the replacement of inherent structures with a grid structure. Throwing away meaningful structures only to attempt to rediscover them using methods such as Raster-to-Vector [9], seems counter intuitive. The use of more authentic and efficient representations have so far been hampered by the lack of powerful machine learning algorithms that are able to handle input of arbitrary size, shape, and connectivity [10]. This is quickly changing due to advances in the emerging class of DNNs called Graph Neural Networks (GNN).

1.1. Contribution

We use GNNs for floor plan analysis, thereby avoiding fundamental problems plaguing raster-based methods. We address the lack of suitable datasets by introducing a new public dataset that consists of distinct buildings and uniquely supports both rasterized and graph-based methods. The contributions can be summarized as follows:

- Comparison of state-of-the-art GNNs performing node classification in graphs from CAD floor plans.

- Demonstration of the performance and accuracy benefits of a graph-based approach to floor plan analysis.

- The Repository of Unique Buildings (RUB) dataset.

2. RELATED WORK

Early work involved image processing techniques such as Macé et al.'s [6] use of the Hough transform for detecting walls and Ahmed et al.'s [7] use of morphological operations for separating text and graphics. More recently, Heras et al. [11] proposed a patch-based segmentation method where sets of features are extracted from image patches before labels are assigned using a bag-of-words model.

Current methods such as those presented by Zhu et al. [5] and Dodge et al. [3] employ Fully Convolutional Networks to segment walls of varying thickness. Following the segmentation, Dodge et al. [3] use the Faster R-CNN object detector to locate individual rooms. Sharma et al. [12] use low and high level semantic features from a CNN to measure floor plan similarity. Zeng et al. [2] use a multi-task CNN with a room-boundary-attention mechanism for floor plan recognition and

room type classification. Sharma et al. [13] propose a conjunction of autoencoder, Cyclic GAN and CNN for mapping between domains and for retrieving similar floor plan image.

The use of graphs in floor plan analysis has been seen before. After morphological operations and connected component analysis, Sharma et al. [8] create graphs of segmented floor plan elements in order to compare room layouts. Yamasaki et al. [14] use a modified McGregor's algorithm to extract a graph model from segmented floor plans, resulting in a maximum common subgraph that is used to measure similarity. Heras et al. [11] generate graphs after segmentating doors, windows, and walls. They then use the A* algorithm to identify the final door and window entities. Most important, Gerstweiler et al. [15] create graphs directly from the lines, arcs, polylines, etc. that make up CAD files, thereby avoiding the problematic transformation to the image domain. Subsequently, openings, walls, and rooms are detected in the graph based on a sets of handcrafted conditions.

With recent breakthroughs in GNNs and the new possibilities they bring, it is worth combining the idea from Gerstweiler et al. [15] with GNNs. GNNs follow a recursive neighborhood aggregation scheme, where feature vectors are passed between neighboring nodes, capturing structural information within nodes' k-hop neighborhood [16]. This encourages similar representation across neighboring nodes and the stacking of graph convolution layers result in progressively higher-level features as seen in CNNs. The power of graph-based deep learning has been demonstrated by Wang et al. [17], who significantly outperform prior work that rely on rasterized representations [18] or does not take advantage of graph structures [19]. Some of the most prominent GNNs include, Graph Convolutional Network (GCN) [20], Graph Attention Network (GAT) [21], GraphSAGE [22], Attention-based Graph Neural Network (AGNN) [23], Topology Adaptive Graph Convolutional Network (TAGCN) [24], and Graph Isomorphism Network (GIN) [16].

3. DATASET

We add to the diversity of publicly available floor plans by making our Repository of Unique Buildings (RUB) available for research in floor plan analysis. The dataset consists of floor plans from two universities and a concert hall. The dataset addresses the lack of support for vectorized floor plan representations in current datasets by providing the source CAD files as well as both rasterized and vectorized representations. RUB consists of 74 floor plans, ranging in scale from smaller buildings rasterized at 500x3,000 pixels to large building complexes taking up 10,000x18,000 pixels. The number of doors in the dataset totals 1744. With our graph representation this corresponds to 21,874 door nodes out of 189,850 total nodes. Figure 2 gives an overview of the tools (rounded rectangles) and data (rectangles) that are made available with RUB. The basis is CAD floor plans in the DXF
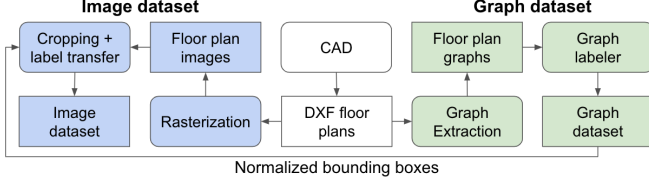
Fig. 2. The Repository of Unique Buildings (RUB) dataset consists of tools (rounded rectangles) and data (rectangles) that support both rasterized and vector-based methods.

format. These can either be rasterized or graphs can be extracted based on CAD primitives. We have labelled the graph nodes and used these to compute and transfer bounding boxes the floor plan images. The result is two comparable datasets, one consisting of node labelled graphs and another of images and bounding boxes. Everything is available at `https://github.com/Chrps/MapGeneralization`.

4. METHOD

4.1. Graph Extraction

Graph nodes and edges are extracted from CAD primitives that are parsed from DXF files using the ezdxf Python library. The conversion from primitives to nodes and edges requires analysis of each CAD primitive. In case of the simple *line* primitive, a node is created for the start point P_0 and another for the end point P_1. The two are connected by an edge. A more complex primitive such as a *circle* is sampled at increments of θ_i based on a center point C and radius r with an angular resolution of $10°$. The angular resolution is determined heuristically in order to faithfully capture the shape of the primitives. The sampling of an *arc* is illustrated in Figure 3. Unlike the *circle*, *arcs* are constrained to a $\Delta\theta$ range.
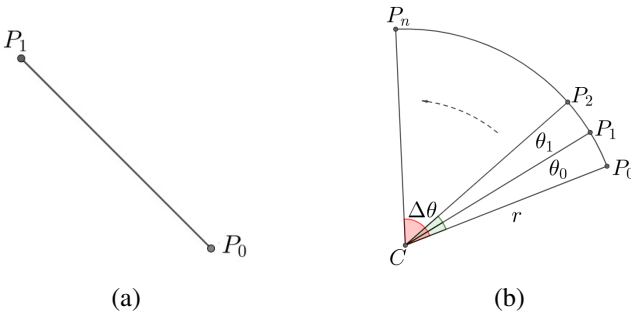
Fig. 3. (a) Nodes are created for P_0 and P_1 and connected by an edge. (b) Points are interpolated on an arc with center C and radius r at increments of θ_i across $\Delta\theta$.

The result of the primitive conversion and sampling is a large number of small disjoint graphs that are subsequently joined by merging any nodes with similar position. Node attributes are computed using the graph and the relationship between

Name	Accuracy[%]	Name	Accuracy[%]
GCN [20]	94.9	GAT [21]	**97.3**
GraphSAGE [22]	94.5	AGNN [23]	87.6
TAGCN [24]	95.9	GIN [16]	95.4

Table 1. Door vs. non-door node classification performance of selected GNN architectures.

neighboring nodes. The attributes include: (1) node degree, (2) max edge angle, (3) min edge angle, (4) max edge length and (5) min edge length. All of the attributes are normalized.

4.2. Node Classification

GNN variants have shown state-of-the-art results on both node and graph classification tasks [16] and look ideally suited for classifying graph nodes as doors or non-door. The novelty of the floor plan dataset and the uncertainty associated with results presented using current benchmarks [10] warrants a comparison between a selection of GNNs. Using the Deep Graph Library we implement the six GNNs mentioned at the end of Section 2. For comparison each network consists of 8 layers and undergo a parameter sweep. The networks are trained using a weighted cross-entropy loss for 500 epochs. Table 1 gives an overview of the performance and shows that performance is generally in line between GNN variants, with GAT achieving a slight lead.
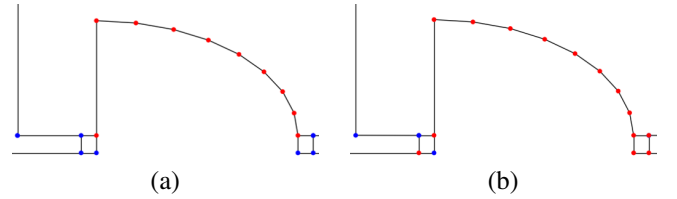
Fig. 4. (a) Ground truth labelling. (b) Labelling by GAT GNN. Door nodes are red and non-door nodes are blue.

Figure 4 (a) shows a fragment of a graph where nodes are labelled according to our ground truth. Figure 4 (b) shows the same fragment with nodes labelled according to the predictions from the GAT GNN. Notice that some non-door nodes in close proximity to door nodes have been misclassified. This is generally not a big problem but may result in less accurate localization in the generalization step that follows. Worst case is nearby doors being registered as one.

4.3. Generalization from Node Labels

Generalization is the process of replacing diverse representations with a general symbol. Here, we use bounding boxes as placeholders. Graph nodes are turned into axis aligned bounding boxes by first removing all non-door nodes from the graphs. This results in a number disjointed sub-graphs. These

	Name	Precision[%]	Recall[%]	Accuracy[%]
our	**GAT**	**75.2**	**75.9**	**60.8**
	GAT	96.7	97.6	94.5
	Dodge et al. [3]	**0.0**	**0.0**	**0.0**
	Dodge et al. [3]	84.2	65.6	58.4
other	Dodge et al. [3]	-	-	96.0
	Liu et al. [9]	91.9	90.2	-
	Yama. et al. [14]	-	-	65.8
	Zeng et al. [2]	-	-	86.0

Table 2. (Top) Performance comparison between our GNN method and Dodge et al. [3] on the RUB dataset. **Bold** is used to highlight results with a strict $IoU > 0.95$ criteria (otherwise $IoU > 0.5$). (Bottom) Performance as reported in related work on other datasets. Results are all based on different subsets of the LIFULL Home's database.

sub-graphs are further separated using the DB-scan [25] clustering algorithm. Finally, the maximum and minimum along each of the two major axes define the bounding boxes for each sub-graph. This procedure is performed on both ground truth labelled graphs and on graphs labelled by the GNN. An example of the resulting bounding boxes is shown in Figure 5

5. RESULTS

Positional accuracy is very important in this application. For this reason, performance is measured using a very strict Intersection-over-Union (IoU) criteria. A performance comparison between our GAT GNN and Dodge et al. [3] is presented in the top half of Table 2. The bottom half shows performance as reported in existing works for their respective datasets. The results from related work are all based on different datasets and since it is not possible to apply our method to existing raster-datasets, these results simply serve as an indicator for the overall level of performance that can be expected. With an accuracy of 96.0% Dodge et al. [3] looks like the best performing method. For this reason Dodge et al. [3], which detects doors using the Faster R-CNN object detector, is used as our baseline. Liu et al. [9] report 91.9% precision and 90.2% recall for their openings category with $IoU > 0.5$. Yamasaki et al. [14] and Zeng et al. [2] both perform pixel level segmentation. Yamasaki et al. [14] achieves 65.8% accuracy for doors and Zeng et al. [2] achieves 86.0% accuracy in a combined door and window category. Even under a strict localization requirement of $IoU > 0.95$ our graph-based method achieves a precision of 75.2% and a recall of 75.9%. The raster-based detector is unable to detect doors when subjected to the $IoU > 0.95$ requirement. For this reason we also report results with a more relaxed $IoU > 0.5$. Here, the graph-based method achieves a precision of 96.7% and a recall of 97.6%, while our implementation of Dodge et al. [3] (Faster R-CNN) achieves a precision of 84.2% and a recall of 65.6%.

(a)
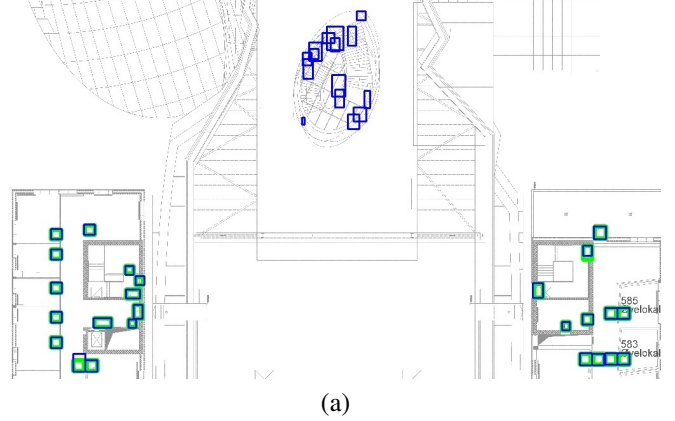
Fig. 5. Crop from a challenging example from our validation set, which highlights the high level of performance for conventional layouts and difficulties with unusual architecture. Green boxes correspond to the ground truth and blue boxes correspond to detections by the GNN.

The drop from 96.0% to 58.4% accuracy when Dodge et al. [3] is applied to our dataset shows that the proposed dataset is significantly more challenging than the raster-datasets currently in use. The reason for this is clearly visible in Figure 5, where our GNN performs well in the conventional areas of the building, accurately identifying and localizing individual doors. Meanwhile, it fails in some of the unusual parts of the architecture resulting in many false positives.

6. CONCLUSION

We have demonstrated the benefits of using graph-based representations and GNNs for analysing floor plans. Unlike the baseline Faster R-CNN object detector our graph-based detector is able to cope with very strict localization requirements. The efficient graph-based representation also makes it easy to operate on huge floor plans. When rasterized, large buildings may require 10,000x18,000 pixels or more to represent their footprint, making it necessary to use a window scheme to perform detection using a CNN. The 37.6 percentage point drop in performance for the baseline method when applied to our dataset suggests that our novel dataset is indeed more challenging. We hope to support further comparison and development by publishing our Repository of Unique Buildings (RUB) dataset, which supports both graph and raster-based approaches.

7. REFERENCES

[1] P. Lanjudkar, "Global indoor positioning and indoor navigation market – industry analysis and forecast (2018-2025)," 2018, [accessed: 28 September, 2020].

[2] Z. Zeng, X. Li, Y. K. Yu, and C.-W. Fu, "Deep floor plan recognition using a multi-task network with room-boundary-guided attention," in *Proc. IEEE Int. Conf. on Computer Vision*, 2019, pp. 9096–9104.

[3] S. Dodge, J. Xu, and B. Stenger, "Parsing floor plan images," in *Proc. 15th Int. Conf. on Machine Vision Applications*, 2017, pp. 358–361.

[4] A. Kalervo, J. Ylioinas, M. Häikiö, A. Karhu, and J. Kannala, "Cubicasa5k: A dataset and an improved multi-task model for floorplan image analysis," in *Proc. 21st Scand. Conf. on Image Analysis*, 2019, pp. 28–40.

[5] R. Zhu, J. Shen, X. Deng, M. Walldén, and F. Ino, "Training strategies for cnn-based models to parse complex floor plans," in *Proc. 9th Int. Conf. on Software and Computer Applications*, 2020, pp. 11–16.

[6] S. Macé, H. Locteau, E. Valveny, and S. Tabbone, "A system to detect rooms in architectural floor plan images," in *Proc. 9th IAPR Workshop on Document Analysis Systems*, 2010, pp. 167–174.

[7] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel, "Automatic room detection and room labeling from architectural floor plans," in *Proc. 10th IAPR Int. Workshop on Document Analysis Systems*, 2012, pp. 339–343.

[8] D. Sharma, C. Chattopadhyay, and G. Harit, "A unified framework for semantic matching of architectural floorplans," in *Proc. 23rd Int. Conf. on Pattern Recognition*, 2016, pp. 2422–2427.

[9] C. Liu, J. Wu, P. Kohli, and Y. Furukawa, "Raster-to-vector: Revisiting floorplan transformation," in *Proc. 16th Int. Conf. on Computer Vision*, 2017, pp. 2195–2203.

[10] Z. Wu, S. Pan, F. Chen, G. Long, C. Zhang, and S. Y. Philip, "A comprehensive survey on graph neural networks," *IEEE Trans. on Neural Networks and Learning Systems*, vol. 32, no. 1, pp. 4–24, 2021.

[11] L.-P. de las Heras, S. Ahmed, M. Liwicki, E. Valveny, and G. Sánchez, "Statistical segmentation and structural recognition for floor plan interpretation," *Int. J. on Document Analysis and Recognition*, vol. 17, no. 3, pp. 221–237, 2014.

[12] D. Sharma, N. Gupta, C. Chattopadhyay, and S. Mehta, "Daniel: A deep architecture for automatic analysis and retrieval of building floor plans," in *Proc. 14th Int. Conf. on Document Analysis and Recognition*, 2017, pp. 420–425.

[13] D. Sharma, N. Gupta, C. Chattopadhyay, and S. Mehta, "A novel feature transform framework using deep neural network for multimodal floor plan retrieval," *Int. J. on Document Analysis and Recognition*, vol. 22, no. 4, pp. 417–429, 2019.

[14] T. Yamasaki, J. Zhang, and Y. Takada, "Apartment structure estimation using fully convolutional networks and graph model," in *Proc. ACM Workshop on Multimedia for Real Estate Tech*, 2018, pp. 1–6.

[15] G. Gerstweiler, L. Furlan, M. Timofeev, and H. Kaufmann, "Extraction of structural and semantic data from 2d floor plans for interactive and immersive vr real estate exploration," *Technologies*, vol. 6, no. 4, pp. 101, 2018.

[16] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?," *arXiv:1810.00826*, 2018.

[17] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *Trans. On Graphics*, vol. 38, no. 5, pp. 1–12, 2019.

[18] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 922–928.

[19] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proc. IEEE Conf. on Computer Vision and Pattern Recognition*, 2017, pp. 652–660.

[20] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," *arXiv:1609.02907*, 2016.

[21] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *6th Int. Conf. on Learning Representations*, 2018.

[22] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, 2017, pp. 1024–1034.

[23] K. K. Thekumparampil, C. Wang, S. Oh, and L.-J. Li, "Attention-based graph neural network for semi-supervised learning," *arXiv:1803.03735*, 2018.

[24] J. Du, S. Zhang, G. Wu, J. M. Moura, and S. Kar, "Topology adaptive graph convolutional networks," *arXiv:1710.10370*, 2017.

[25] M. Hahsler, M. Piekenbrock, and D. Doran, "dbscan: Fast density-based clustering with R," *J. of Statistical Software*, vol. 91, no. 1, pp. 1–30, 2019.