FROM TOPOLOGY TO SPATIAL INFORMATION: A COMPUTATIONAL APPROACH FOR GENERATING RESIDENTIAL FLOORPLANS

MOHAMED EL MESAWY[1], NAWAL ZAHER[2] and AHMED EL ANTABLY[3]
[1,2,3]*Arab Academy for Science, Technology and Maritime Transport.*
[1]*m.elmesawy50@student.aast.edu, 0009-0005-2268-7928*
[2]*nawalzaher@aast.edu, 0000-0001-7984-8060*
[3]*ahmed.antably@aast.edu, 0000-0002-0151-0809*

Abstract. Multimodal models that combine different media like text, image, audio, and graph have revolutionised the architectural design process, which could provide automated solutions to assist the architects during the early design stages. Recent studies use Graph Neural Networks (GNNs) to learn topological information and Convolution Neural Networks (CNNs) to learn spatial information from floorplans. This paper proposes a deep learning multimodal model incorporating GNNs and the Stable Diffusion model to learn the floorplan's topological and spatial information. The authors trained a Stable Diffusion model on samples from the RPLAN dataset. They used graph embedding for conditional generation and experimented with three approaches to whole-graph embedding techniques. The proposed Stable Diffusion model maps the user input, a graph representing the room types and their relationships, to the output, the predicted floorplans, as a raster image. The Graph2Vec and contrastive learning methods generate superior representational capabilities and yield good and comparable results in both computationally derived scores and evaluations conducted by human assessors, compared to the Graph Encoder-CNN Decoder.

Keywords. Floorplan Generation, Deep Generative Models, Multimodal Machine Learning, Graph Neural Networks [GNNs], Representation Learning.

1. Introduction

In recent years, advancements in artificial intelligence (AI) have brought renewed attention to the role of Computer-Aided Architectural Design (CAAD). The use of AI in architectural design has proliferated from the early days of CAAD to the current visions of man-machine symbiosis. This growth relies on computer hardware and software advances and requires a shift in architects' design thinking paradigms. It necessitates collaboration among professionals from multiple domains, including architects, computer scientists, data scientists, and machine learning engineers. This

ACCELERATED DESIGN, Proceedings of the 29th International Conference of the Association for Computer-Aided Architectural Design Research in Asia (CAADRIA) 2024, Volume 1, 129-138.

paper proposes a deep-learning multimodal model incorporating GNNs and diffusion models to learn topological and spatial information from architectural floorplans. The aim is to develop a system that generates high-quality architectural floorplans to assist architects during the early design stages.

The significance of this paper lies in several key aspects. Firstly, the authors employ the whole graph embedding technique to capture and represent a floorplan's intricate structure and topology. This approach enables a comprehensive understanding of the floorplan spatial relationships and design elements. Secondly, the proposed use of the diffusion model for floorplan generation demonstrates its superior performance compared to Generative Adversarial Networks (GANs) regarding image generation quality. Lastly, this study encompasses multiple approaches to obtain whole-graph embeddings as an agnostic task. These embeddings have broader implications beyond floorplan generation, as they can be utilised in various downstream tasks such as graph classification and conditional generation. This versatility enhances our research's applicability and potential impact in diverse architectural and computational design domains.

2. AI and Floorplan Generation

The recent growth in the use of Artificial Neural Networks (ANNs) in computational design reflects the fast advancement in research in generative models (Dhondse A et al., 2020), the increase in computational power, and the availability of training datasets (Hodas & Stinis, 2018) such as RPLAN, CubiCasa5K, and CubiGraph5K. Moreover, using many computational algorithms and variants of neural networks based on graphs, such as Graph Convolutional Networks (GCN) (Carta, 2021), has led to considerable advancements in graph processing and generation. In general, algorithms based on graph theory result in quite an effective manipulation of data in spatial configurations. Designers and those trained in spatial abstraction usually find topological approaches intuitive, for they have data structures like bubble diagrams and direct applications to spatial organization. Wassim Jabi's work innovatively employs topological graphs using the Topologic plugin, presenting a notable advancement for building classification tasks (Alymani et al., 2023).

The literature may be divided into three categories of floorplan conditional generative methods based on the input type: pixel-based, language-based, and graph-based approaches. Moreover, generative deep learning models typically comprise two main components: the encoder and the decoder. Combining different modalities within the encoder and decoder exposes different generative models utilising different media types, each with advantages and disadvantages for floorplan generation tasks. Some approaches may employ a pre-processing step to convert the input from one type to another before feeding it into the model or even post-processing the output.

First, following a pixel-based approach, some researchers developed their model using the Pix2Pix model, which is a version of Generative Adversarial Networks (GANs) that takes an image representing the floorplan footprint as input and a floorplan raster image as output (Figure 1). The model consists of convolutional parts in both the encoder and decoder. This approach's advantage is utilising convolutional layers in the model decoder. CNNs are critical in preserving and leveraging spatial information during the model's training and prediction phases. They extract spatial information

from an image by performing localised computations using convolution kernels, which generate floorplans that look like real images and make much sense from an architectural perspective. The limitation of this approach is that the user has no control over the floorplan program as it is solely conditioned on the floorplan footprint and the door location. Recently, Veloso et al. tried to address this limitation by converting the input graph with the room areas to a bubble diagram image, then dealing with the input bubble diagram image and output floorplan image as Pix2Pix (Veloso et al., 2022). However, while the input in the graph format has the floorplan topological information, the conversion of the graph into a pixelated image does not retain this information explicitly as in the graph format. As a result, the model may easily generate unrealistic designs (Figure 1).

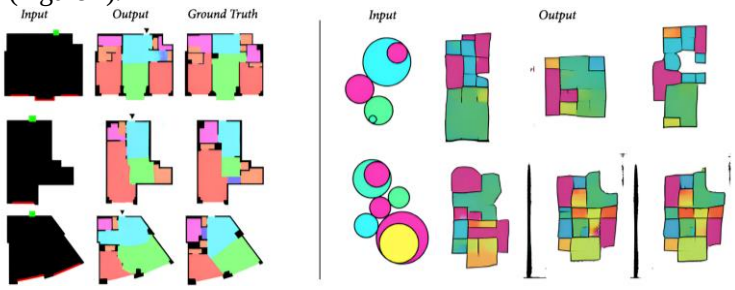

Figure 1. Left: Success example of using Pix2Pix in architectural floorplans (source: Chaillou, 2020). Right: Examples of overlapping discs after converting the graph and zones to an image (source: Veloso et al., 2022).

Second, in the graph-based approach, the user's input is a graph containing the room types and their edge relationships, fed to GNNs in the encoder part (Nauata et al., 2020, 2021). The generated output of this approach is a series of bounding boxes representing each room, which are later combined to generate the final floorplan during the post-processing phase. The advantage of this model is rooted in the GNN's ability to learn the underlying topology of the input graph. This model leverages the graph structure as a source of non-spatial information that captures the connectivity patterns between the nodes (rooms). However, this model's limitation lies in its decoder component, which treats the output as a regression task by predicting the bounding boxes of each room separately. Consequently, there is a risk of some rooms overlapping, resulting in generated floorplans that do not make architectural sense regarding rooms' dimensions and spatial qualities, as shown in the failure cases in Figure 2.

Figure 2. Left: Overlapping bounding boxes of rooms in House GAN.
Right: Failure and success examples of the generation (source: Nauata et al., 2020).

Third, in the language-based approach, the language model encoder learns the mapping between the textual input and the corresponding floorplan generated by the decoder. The decoder may also predict bounding boxes (Galanos et al., 2023) (Figure 3) or generate images from the text as done in generative models such as Stable Diffusion, GAN text, Dall-2, and Mid-Journey. Nevertheless, the models that generate the room output as bounding boxes may lack spatial information.

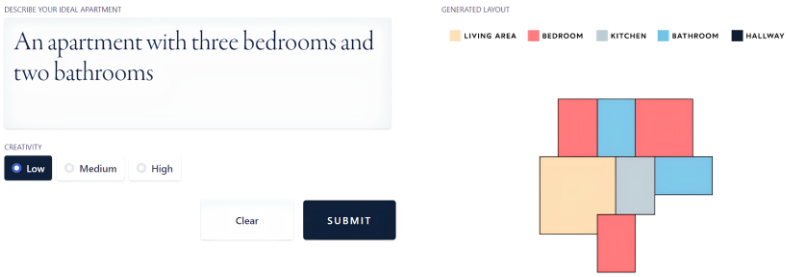

Figure 3. Architex user interface and its raw prediction (source: Ramesh et al., 2022).

To this end, this paper proposes a deep-learning multimodal model that combines the strengths of GNN and CNN in Stable Diffusion to generate a high-quality raster image of the floorplan. Specifically, it employs GNN in the encoder component to capture the graph input's topological non-spatial information and the decoder's convolutional layers to preserve the spatial characteristics of the generated floorplan. In other words, the paper replaces the text encoder part in the original Stable Diffusion model with a graph encoder pre-trained as whole-graph embedding.

We have approached the graph encoder as an architectural language model by training it as an agnostic task to learn the whole-graph embedding, which holds the floorplan semantics. The learned embeddings can then be utilised in downstream tasks such as graph classification or floorplan generation.

3. Methods

The paper uses a three-stage method: data preparation, generative model, and finally, post-processing.

3.1. DATASET PREPARATION AND PRE-PROCESSING

The RPLAN dataset (Wu et al., 2019) is a collection of real-world residential buildings containing over 80k floorplans with 13 room types (Hu et al., 2020). We split 1000 floorplans from the RPLAN dataset into 800 and 200 samples for the train and test sets, respectively. We adopted the same colouring labels and room mappings from the pre-processing stage (Rodrigues et al., 2021), as the colours have high contrast, which helps in the room's segmentation. We mapped the dataset room types to only six: public area, room, storage, kitchen, bathroom, and balcony. In the pre-processing step, we extracted the graph and its corresponding floorplan image from the raw multichannel floorplan file provided in the dataset. Figure 4 shows an example of the extracted geomatic information from the structured floorplan, which is stored in a serializable format to generate a graph that describes the room relation in a floorplan.

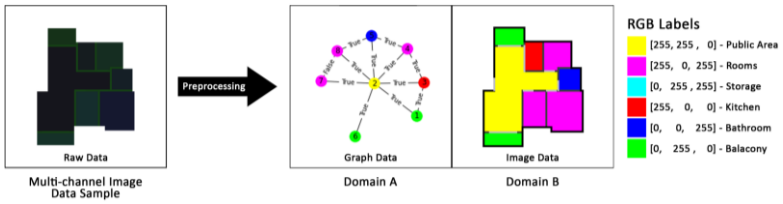

Figure 4. The pre-processing stage: extracting the graph and the floorplan image-RPLAN Dataset.

3.2. MODEL TRAINING

First, we trained the graph encoder as an agnostic task to get the whole-graph embeddings. Then, we used these embeddings in the downstream task of floorplan generation. The whole-graph embeddings represent graph structures in a lower-dimensional space (Figure 5). The aim is to encode the graphs (reduced bubble diagram) such that the similarity in the embedding space approximates the similarity in the original network. After separately training the graph encoder, we replaced the text encoder in the original Stable Diffusion model with the pre-trained graph encoder. Finally, we tested three implementations for the graph encoder in conjunction with the Stable Diffusion model, where we trained the diffusion model from scratch with each graph encoder. We discuss the graph encoders in the following sections.

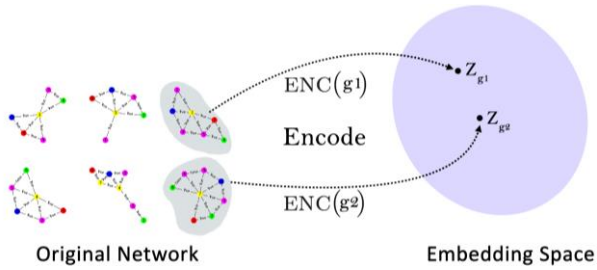

Figure 5. Representation learning on networks: whole graph embedding.

The first graph encoding method is Graph2vec (Narayanan et al., 2017) for whole-graph embedding. This method has received attention recently due to its ability to capture graph structure information and generate embeddings usable in various downstream tasks. The method uses a skip-gram model to learn vector representations of subgraphs from a given graph. These vector representations are combined to generate an embedding for the entire graph (Figure 5). We used the off-the-shelf Graph2vec algorithm implemented in Karate Club (Rozemberczki et al., 2020).

The second approach is borrowed from the Contrastive Language-Image Pre-training (CLIP) model (Radford et al., 2021). One of the key features of CLIP is its use of contrastive learning, a technique that allows the model to learn by contrasting similar and dissimilar pairs of data, text, and images. Using a contrastive loss function, CLIP utilises a transformer-based model pre-trained on a large corpus of images and text. The model may then be finetuned for specific downstream tasks, such as image classification, object detection, and image captioning. For this paper, we applied the concept of contrastive learning by replacing the text encoder with a graph encoder and

used a pre-trained image encoder, which we trained first within a convolutional autoencoder model Figure 6. In the next training step, only the graph encoder is learnable by contrastive learning between the graph and its corresponding floorplan raster image, as shown in Figure 7.

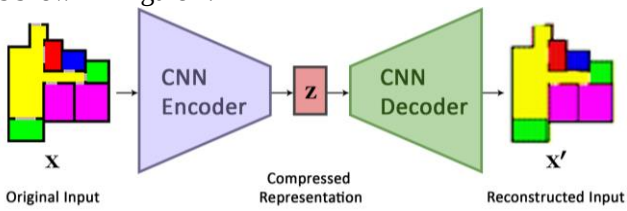

Figure 6. Learning phase of the CNN encoder and decoder components using a convolutional autoencoder model.

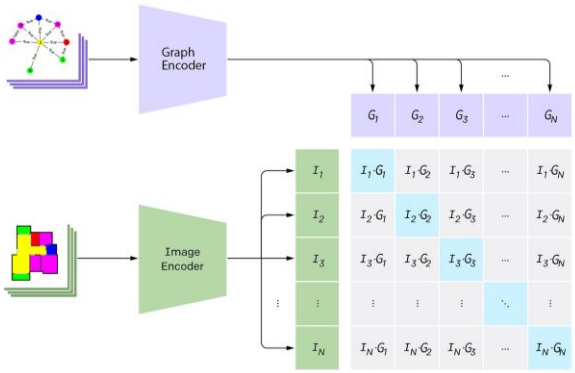

Figure 7. Contrastive learning between graph and images (adapted from: Radford et al., 2021)

The third method for training the graph encoder is the Graph Encoder-CNN Decoder. This method replaces only the convolution encoder part from the convolutional autoencoder model (Figure 6) with a graph encoder. The original CNN decoder is trained with a graph encoder consisting of stacked GNN layers, particularly Graph Sage (Hamilton et al., 2018) (Figure 8). The model learns the representation of the graph input data by compressing it into a lower-dimensional latent space and then constructing the corresponding floorplan image from this compressed representation. The model is trained by minimising the difference between input and constructed data. This model could be used as an end-to-end approach to generate the floorplan image from an input graph. However, we only used the trained graph encoder part in the downstream floorplan generation task.

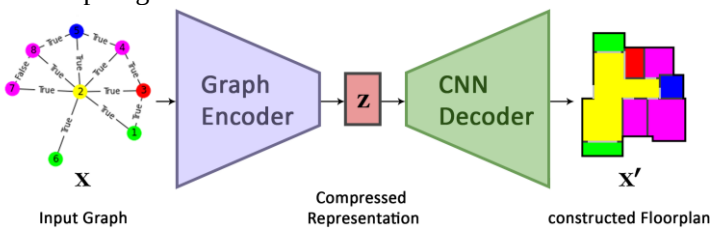

Figure 8. Graph encoder/CNN decoder.

3.3. VISUALISING THE GRAPH EMBEDDINGS

After training the graph encoder separately to learn the embeddings of the graphs corresponding to the floorplans in the training set, we visualised the output graph embeddings with the corresponding floorplan images using the Embedding Projector in Tensorboard. To visually assess the effectiveness of the learned embeddings, it is possible to randomly select a sample of floorplans and evaluate the K-nearest similar plans according to the graph embeddings. An effective graph encoder model should embed similar graphs with similar embeddings to be closer and dissimilar graphs farther apart in the latent space.

3.4. DIFFUSION MODEL TRAINING

We trained the Stable Diffusion model on a single NVidia P100 GPU using PyTorch implementation. Our Stable Diffusion model consists of the pre-trained graph encoder and the image generator component (Figure 9). The training parameters of the Stable Diffusion model are 1000 noising steps, an image size of 64x64 pixels for faster training, training epochs of 1500 epochs with a batch size of 8, and a learning rate of 0.0003. Each of the three trained models took 12 hours to train with the parameters mentioned above.

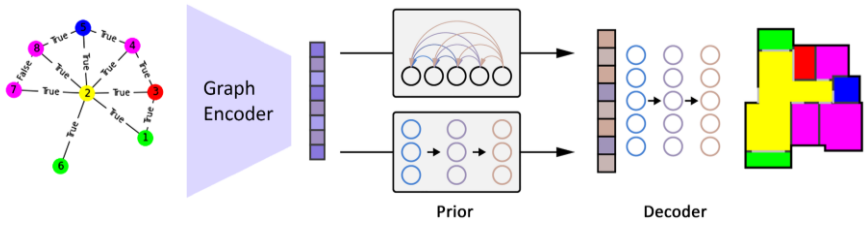

Figure 9. Proposed Stable Diffusion: graph-image mapping (adapted from: Ramesh et al., 2022).

3.5. POST-PROCESSING

We employed Real-ESRGAN (Wang et al., 2021), a super-resolution model, to enhance the generated lower-resolution images by up-sampling the output four times from 64x64 pixels to 256x256 pixels. This step allowed us to obtain higher-quality floorplan images without having to perform training on larger images in the Stable Diffusion model (Figure 10).

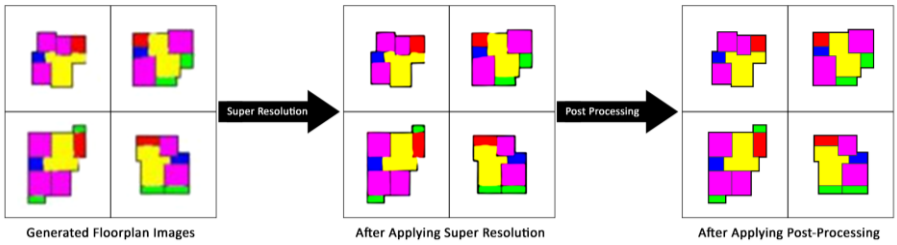

Figure 10. The post-processing step using the super-resolution model to enhance the quality of the generated image, then applying wall alignment.

4. Results and Discussion

We evaluated the three versions of the generative models, denoted as Model A, Model B, and Model C, using three metrics focusing on image generation performance: Fréchet Inception Distance (FID), Intersection over Union (IoU) and survey questionnaire.

FID is a metric used to evaluate the quality of generated images by comparing them to a set of real images. It is a widely used measure in the field of GANs and generative models in general (Heusel et al., 2018). A lower FID score indicates that the generated images are of high quality and diversity. However, an FID score for architectural floorplan generation does not consider essential architectural criteria, such as the conditional program of the input. Hence, we developed a method for calculating the Intersection over Union (IoU) between the program of the output-generated floorplan and the ground truth. The analysis of variance (ANOVA) test, applied to the IoU metric, indicated a significant difference among the models ($F=51.95$, $p<0.001$). Subsequently, we used a post-hoc analysis to explore specific group differences. The post-hoc analysis, with all p-values less than 0.001, revealed significant differences between all pairs of models, reinforcing the ANOVA findings.

Besides the above computational methods, we developed a survey questionnaire to assess the quality of the generated floorplans from a human perspective. We used a form containing 15 samples of generated floorplans, five from each model in random order, populating 60 random forms, each with 15 different floorplans selected from the test set. Sixty architectural students responded to the survey. Employing repeated measures ANOVA to analyse students' responses indicated a statistically significant ($F=67.73$, $p<0.001$) difference between the models. A subsequent post-hoc analysis ($p<0.001$) showed a significant difference between Models A and C, as well as B and C, but none between A and B.

Table 1: compares the FID, IoU and the survey questionnaire for each model on the test set.

Model	Graph Encoder	Graph Enc. Training Time	SD Training Time	FID	IoU		Human Eval.	
					mean	std	mean	std
A	Graph2Vec	1 min.	12 hours	66.9	0.88	0.12	3.60	1.19
B	Contrastive Learning	8 hours	12 hours	72.1	0.8	0.12	3.70	1.18
C	Graph Enc. - CNN Dec.	12 hours	12 hours	134.5	0.69	0.28	2.35	1.3

From a qualitative point of view, the Stable Diffusion model generates better floorplans in terms of the rooms' architectural qualities when conditioned with a pre-trained whole-graph embedding (Figure 11). This paper's Graph2Vec and contrastive learning methods generate superior representational capabilities compared to alternative graph encoders and have close scores across the FID, IoU, and human evaluation. On the other hand, the Graph Encoder-CNN Decoder produced the least favourable results in terms of FID, IoU, and human visual inspection. Interestingly, the computational scores are close to human evaluation in terms of generation quality.

Generally, three factors influence the quality of the generated output: the quality of the pre-trained graph embeddings, the length of the embedding vectors, and the generative model itself. First, the quality of the pre-trained graph embeddings is a limitation of the Stable Diffusion generation. In this aspect, the Graph2Vec and the graph encoder based on contrastive learning outperform the third method, hence generating floorplans from the Stable Diffusion model that are highly similar to the ground truth. The second limitation is the length of the embedding vectors. Generally, higher dimensional latent space can capture more information to some extent. Last, the generative model itself, especially the design of the loss function, penalises the model for generating outputs that lack architectural coherence. Future investigations may consider diversifying the types of datasets used for training to enhance the model's adaptability and exploring the model's applicability in non-residential or diverse cultural contexts.

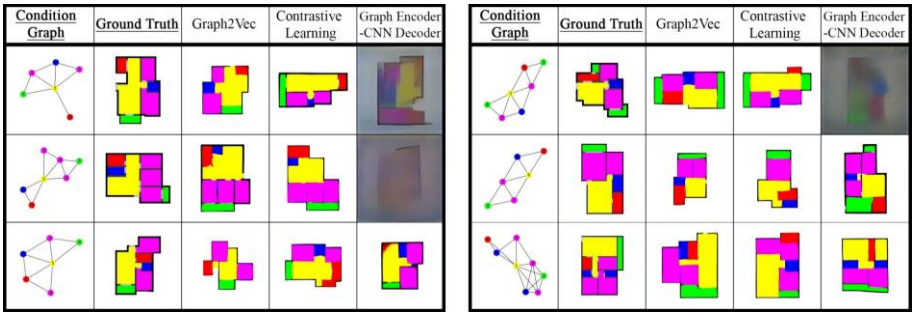

Figure 11. A comparison between the generated floorplans and the ground truth for the same input graph using the three proposed methods.

5. Conclusion

This research proposes a deep-learning multimodal architecture that uses GNN and the Stable Diffusion model to learn the topological and spatial information of architectural floorplans. Eight hundred samples from the RPLAN dataset were used for training and 200 for testing. Three different evaluation metrics show that our proposed method, which replaces the text encoder with a pre-trained graph encoder, generates high-quality architectural floorplans. However, the quality of the graph embeddings, the length of the embedding vectors, and the architecture of the generative model influence the quality of the generated output. The Graph2Vec and the graph encoder based on contrastive learning methods produced superior representations compared to the third graph encoder. Overall, the proposed method has significant potential to assist architects in the early design stages, enabling them to select the most suitable floorplan for their design needs.

References

Alymani, A., Jabi, W., & Corcoran, P. (2023). Graph machine learning classification using architectural 3D topological models. SIMULATION, 99(11), 1117–1131. https://doi.org/10.1177/00375497221105894

Carta, S. (2021). Self-Organizing Floor Plans. Harvard Data Science Review. https://doi.org/10.1162/99608f92.e5f9a0c7

Chaillou, S. (2020). ArchiGAN: Artificial Intelligence x Architecture. In P. F. Yuan, M. Xie, N. Leach, J. Yao, & X. Wang (Eds.), Architectural Intelligence (pp. 117–127). Springer Nature Singapore. https://doi.org/10.1007/978-981-15-6568-7_8

Dhondse A, Kulkarni S, Khadilkar K, Kane I, Chavan S, Barhate R, & 3rd International Conference on Data Management, A. and I., ICDMAI 2019. (2020). Generative Adversarial Networks as an Advancement in 2D to 3D Reconstruction Techniques. Adv. Intell. Sys. Comput. Advances in Intelligent Systems and Computing, 1016, 343–364.

Galanos, T., Liapis, A., & Yannakakis, G. N. (2023). Architext: Language-Driven Generative Architecture Design (arXiv:2303.07519). arXiv. http://arxiv.org/abs/2303.07519

Hamilton, W. L., Ying, R., & Leskovec, J. (2018). Inductive Representation Learning on Large Graphs (arXiv:1706.02216). arXiv. http://arxiv.org/abs/1706.02216

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2018). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium (arXiv:1706.08500). arXiv. http://arxiv.org/abs/1706.08500

Hodas, N. O., & Stinis, P. (2018). Doing the Impossible: Why Neural Networks Can Be Trained at All. Frontiers in Psychology, 9, 1185. https://doi.org/10.3389/fpsyg.2018.01185

Hu, R., Huang, Z., Tang, Y., Van Kaick, O., Zhang, H., & Huang, H. (2020). Graph2Plan: Learning floorplan generation from layout graphs. ACM Transactions on Graphics, 39(4). https://doi.org/10.1145/3386569.3392391

Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., & Jaiswal, S. (2017). graph2vec: Learning Distributed Representations of Graphs (arXiv:1707.05005). arXiv. http://arxiv.org/abs/1707.05005

Nauata, N., Chang, K.-H., Cheng, C.-Y., Mori, G., & Furukawa, Y. (2020). House-GAN: Relational Generative Adversarial Networks for Graph-constrained House Layout Generation. arXiv:2003.06988 [Cs]. http://arxiv.org/abs/2003.06988

Nauata, N., Hosseini, S., Chang, K.-H., Chu, H., Cheng, C.-Y., & Furukawa, Y. (2021). 02_House-GAN++: Generative Adversarial Layout Refinement Networks. arXiv:2103.02574 [Cs]. http://arxiv.org/abs/2103.02574

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., & Sutskever, I. (2021). Learning Transferable Visual Models From Natural Language Supervision (arXiv:2103.00020). arXiv. http://arxiv.org/abs/2103.00020

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., & Chen, M. (2022). Hierarchical Text-Conditional Image Generation with CLIP Latents (arXiv:2204.06125). arXiv. http://arxiv.org/abs/2204.06125

Rodrigues, R. C., Imagawa, M. K., Koga, R. R., & Duarte, R. B. (2021). Big Data vs Smart Data on the Generation of Floor Plans with Deep Learning. Blucher Design Proceedings, 217–228. https://doi.org/10.5151/sigradi2021-114

Rozemberczki, B., Kiss, O., & Sarkar, R. (2020). Karate Club: An API Oriented Open-source Python Framework for Unsupervised Learning on Graphs (arXiv:2003.04819). arXiv. http://arxiv.org/abs/2003.04819

Veloso, P., Rhee, J., Bidgoli, A., & Ladron de Guevara, M. (2022). Bubble2Floor: A Pedagogical Experience With Deep Learning for Floor Plan Generation. 373–382. https://doi.org/10.52842/conf.caadria.2022.1.373

Wang, X., Xie, L., Dong, C., & Shan, Y. (2021). Real-ESRGAN: Training Real-World Blind Super-Resolution with Pure Synthetic Data (arXiv:2107.10833). arXiv. http://arxiv.org/abs/2107.10833

Wu, W., Fu, X.-M., Tang, R., Wang, Y., Qi, Y.-H., & Liu, L. (2019). Data-driven interior plan generation for residential buildings. ACM Transactions on Graphics, 38(6), 1–12. https://doi.org/10.1145/3355089.3356556