[Supplementary Document] House-GAN: Relational Generative Adversarial Networks for Graph-constrained House Layout Generation

Nelson Nauata[1], Kai-Hung Chang[2], Chin-Yi Cheng[2], Greg Mori[1], and Yasutaka Furukawa[1]

[1] Simon Fraser University
{nnauata, furukawa}@sfu.ca, mori@cs.sfu.ca
[2] Autodesk Research
{kai-hung.chang, chin-yi.cheng}@autodesk.com

The supplementary document provides 1) additional details on the interface utilized for the user study; 2) additional experimental details 3) full architectural specification for CNN-only and GCN baselines; and 4) additional experimental results.

1 User study interface

Figure 1 shows a screenshot of our user study for the realism evaluation. A subject is presented 1) a legend for the room types and their associated colors, 2) a set of ground-truth house layouts as reference, and 3) a pair of generated floorplans for each question. A subject is given 75 question and asked to choose one of the four possible answers ("A is better", "B is better", "Both are good", "Both are bad"), where the entire session takes around 10 to 15 minutes to be completed. The generated sample pair comes from a pair of randomly selected models plus ground-truth. We enforce that each possible pair of model is selected exactly 5 times during the entire session.

2 Additional experimental details

In the compatibility score computation (graph edit distance), we decreased the number of test samples from 5,000 to 1,000 for graphs of size 10-12 and 13+, due to its high computational cost. For all experiments, we compute the graph edit distance score using the function *optimize_edit_distance* from NetworkX version 2.4 [2]. We set the upper bound (max distance) to 40 and added a timeout of 1 hour to this function call. In the ablation study for compatibility, the first baseline without any constraint (i.e. CNN-only without room count, type and connectivity) was evaluated disregarding the room type in the graph edit distance computation, since no room type information is estimated nor given in this baseline.

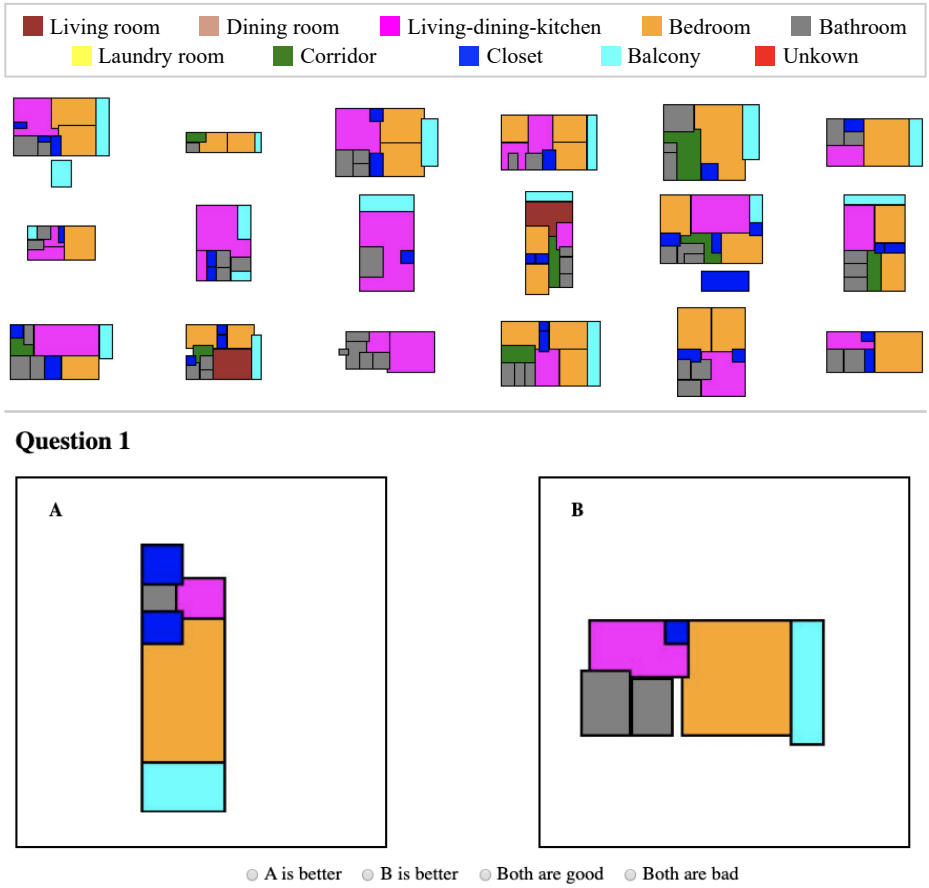

Fig. 1. A screenshot of our user study for the realism evaluation. The legend appears on the top, followed by a set of ground-truth samples in the middle, and a pair of generated sample at the bottom for each question.

3 Baselines architectural specification

Tables 1 and 2 present the full architecture specification for the CNN-only and GCN baselines.

4 Additional results

Figures 2-11 present additional generation results by the proposed system (House-GAN). Each row shows an input bubble diagram and pairs of a generated room mask and the corresponding house layout. We divide the samples into 5 groups depending on the numbers of rooms. We present two pages of results for each group, in a increasing order of the room counts.

Table 1. CNN-only architectural specification. "s" and "p" denote stride and padding. Considering 40 to be the maximum number of rooms in our experiments. "x", "z", "t" and "c" denote the room mask, noise vector, all room types vector concatenated (i.e. 1×400) and all connections (i.e. $780=\binom{40}{2}$) represented as a 1D vector (1×780). Convolution kernels and layer dimensions are specified as ($N_{in} \times N_{out} \times W \times H$) and ($W \times H \times C$).

Architecture	Layer	Specification	Output Size
House layout generator	$concat(z, t, c)$	N/A	1×1308
	$linear\_reshape_1$	1308×1024	$8 \times 8 \times 16$
	$upsample_1$	$16 \times 16 \times 4 \times 4, (s = 2, p = 1)$	$16 \times 16 \times 16$
	$upsample_2$	$16 \times 16 \times 4 \times 4, (s = 2, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_1$	$16 \times 256 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 256$
	$conv\_leaky\_relu_2$	$256 \times 128 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 128$
	$conv\_tanh_1$	$128 \times 40 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 40$
House layout discriminator	$concat(t, c)$	N/A	1×1180
	$linear\_reshape_1$	1180×8192	$32 \times 32 \times 8$
	$concat\_with(x)$	N/A	$32 \times 32 \times 9$
	$conv\_leaky\_relu_1$	$9 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_2$	$16 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_3$	$16 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$downsample_1$	$16 \times 16 \times 3 \times 3, (s = 2, p = 1)$	$16 \times 16 \times 16$
	$downsample_2$	$16 \times 16 \times 3 \times 3, (s = 2, p = 1)$	$8 \times 8 \times 16$
	$conv\_leaky\_relu_1$	$16 \times 256 \times 3 \times 3, (s = 2, p = 1)$	$4 \times 4 \times 256$
	$conv\_leaky\_relu_2$	$256 \times 128 \times 3 \times 3, (s = 2, p = 1)$	$2 \times 2 \times 128$
	$conv\_leaky\_relu_3$	$128 \times 128 \times 3 \times 3, (s = 2, p = 1)$	$1 \times 1 \times 128$
	$reshape\_linear_1$	128×1	1

Table 2. GCN architectural specification. "s" and "p" denote stride and padding. "x", "z" and "t" denote room mask, noise vector, and room type vector. "gcn" layers are implemented using the official code borrowed from Ashual *et al.* [1] and Johnson *et al.* [3]. In "gcn", d_{in} denotes input dimension, d_h hidden dimension and n_l number of layers. All "gcn" layers uses average pooling. Convolution kernels and layer dimensions are specified as $(N_{in} \times N_{out} \times W \times H)$ and $(W \times H \times C)$.

Architecture	Layer	Specification	Output Size
House layout generator	$concat(z, t)$	N/A	1×138
	$linear_1$	138×128	1×128
	gcn	$(d_{in} = 128, d_h = 512, n_l = 2)$	1×128
	$linear_2\_reshape$	128×1024	$8 \times 8 \times 16$
	$upsample_1$	$16 \times 16 \times 4 \times 4, (s = 2, p = 1)$	$16 \times 16 \times 16$
	$upsample_2$	$16 \times 16 \times 4 \times 4, (s = 2, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_1$	$16 \times 256 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 256$
	$conv\_leaky\_relu_2$	$256 \times 128 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 128$
	$conv\_tanh_1$	$128 \times 1 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 1$
House layout discriminator	$linear\_reshape_1(t)$	10×8192	$32 \times 32 \times 8$
	$concat(t, x)$	N/A	$32 \times 32 \times 9$
	$conv\_leaky\_relu_1$	$9 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_2$	$16 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$conv\_leaky\_relu_3$	$16 \times 16 \times 3 \times 3, (s = 1, p = 1)$	$32 \times 32 \times 16$
	$downsample_1$	$16 \times 16 \times 3 \times 3, (s = 2, p = 1)$	$16 \times 16 \times 16$
	$downsample_2$	$16 \times 16 \times 3 \times 3, (s = 2, p = 1)$	$8 \times 8 \times 16$
	$conv\_leaky\_relu_1$	$16 \times 256 \times 3 \times 3, (s = 2, p = 1)$	$4 \times 4 \times 256$
	$conv\_leaky\_relu_2$	$256 \times 128 \times 3 \times 3, (s = 2, p = 1)$	$2 \times 2 \times 128$
	$conv\_leaky\_relu_3$	$128 \times 128 \times 3 \times 3, (s = 2, p = 1)$	$1 \times 1 \times 128$
	gcn	$(d_{in} = 128, d_h = 512, n_l = 2)$	1×128
	$pool\_reshape\_linear_1$	128×1	1

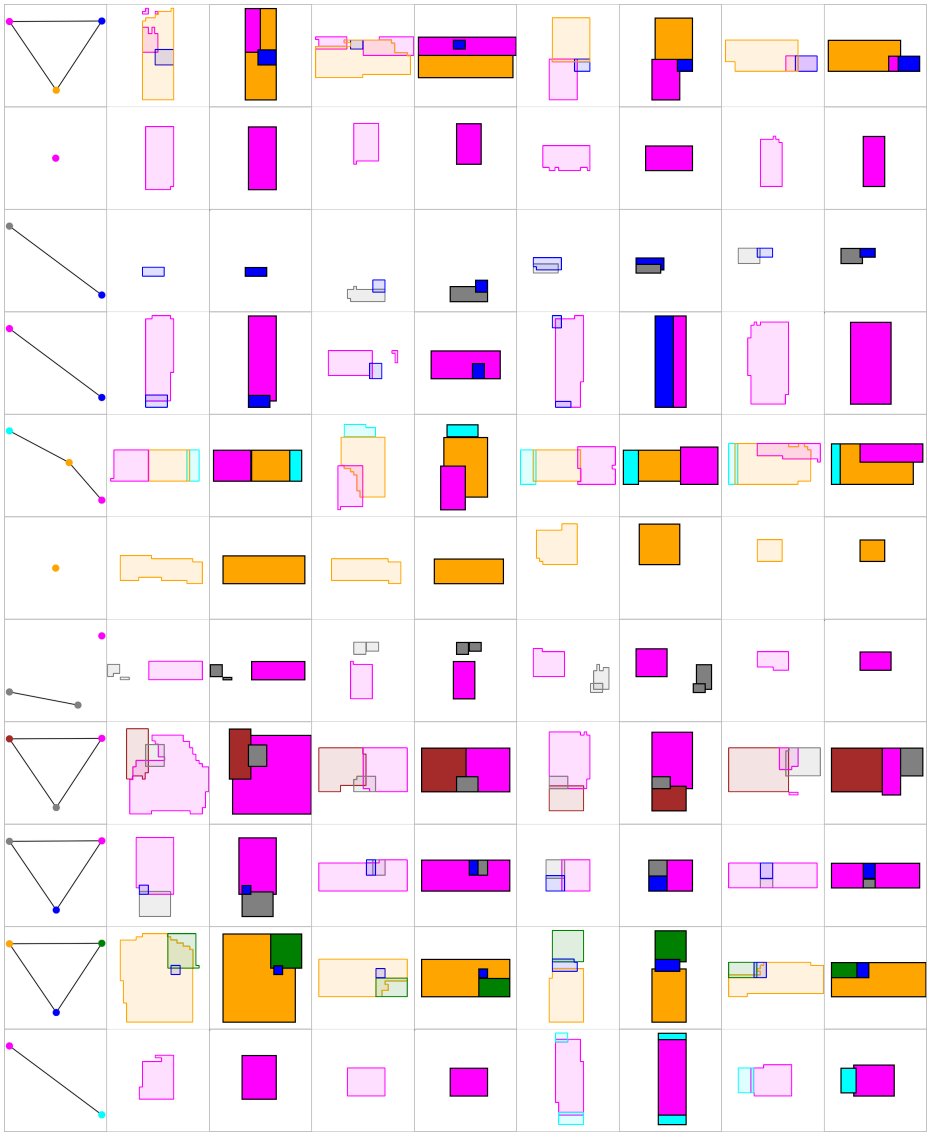

Fig. 2. Additional qualitative results for room masks and final house layouts. Each row shows four generated sample pairs (i.e. mask and layout) for the same graph presented in the first column. Model was trained on graphs of size 4+ and tested on graphs of size 1-3.

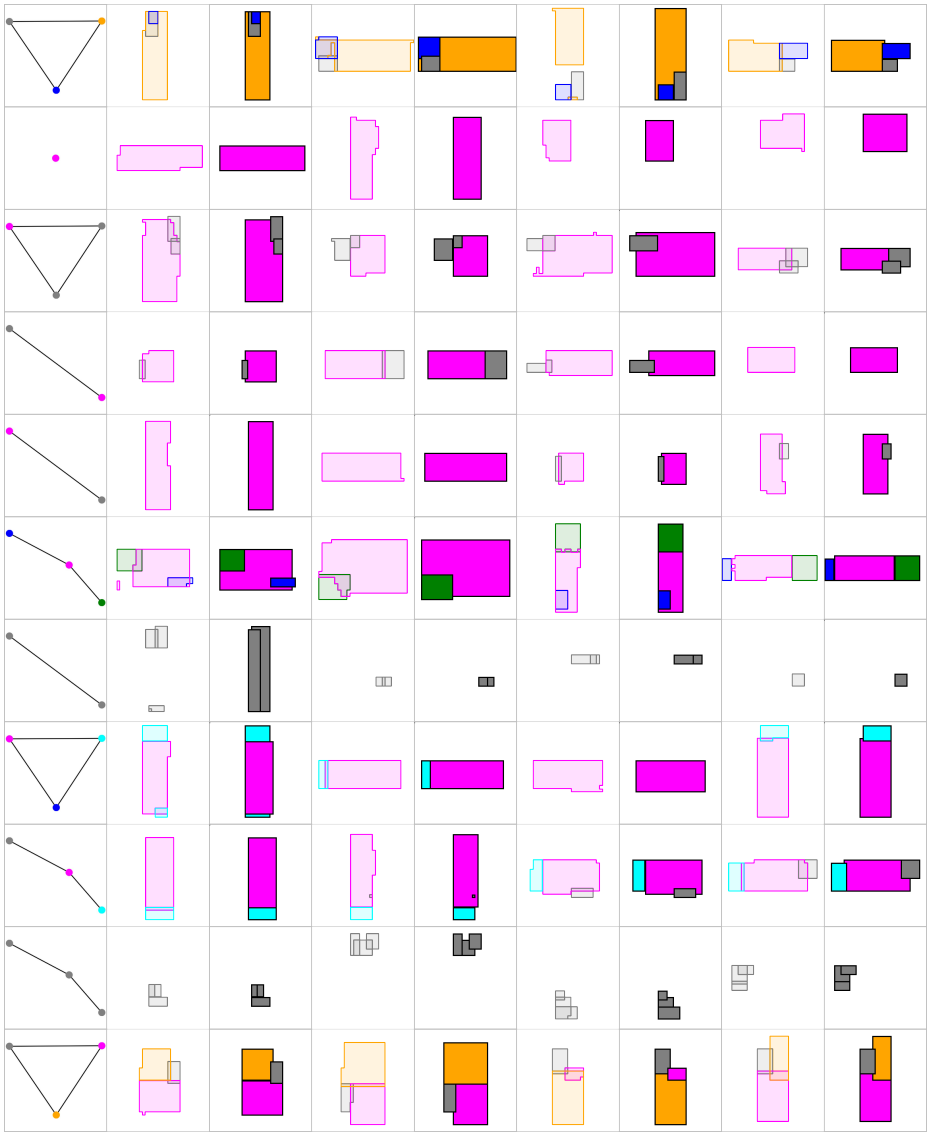

Fig. 3. Continued.

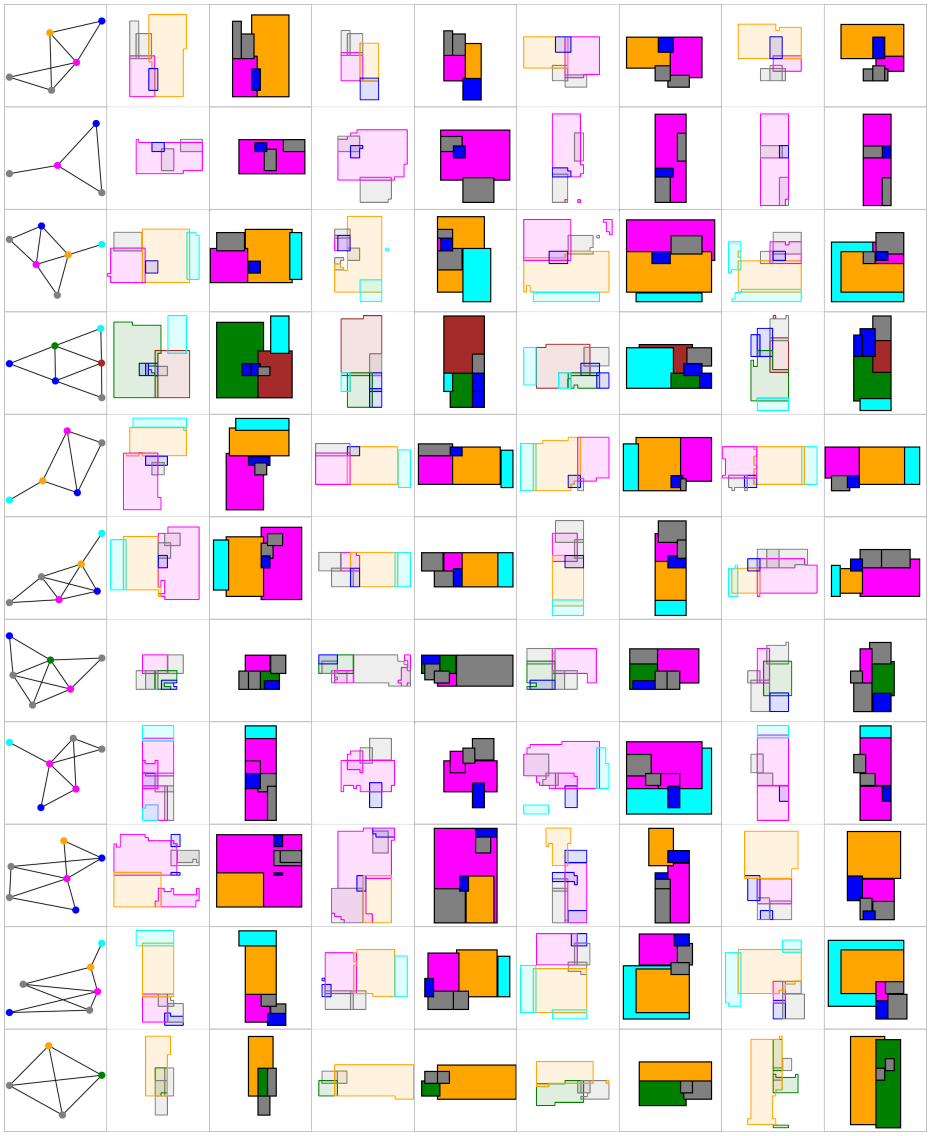

Fig. 4. Results for model trained on graphs of size 1-3 and 7+ and tested on graphs of size 4-6.

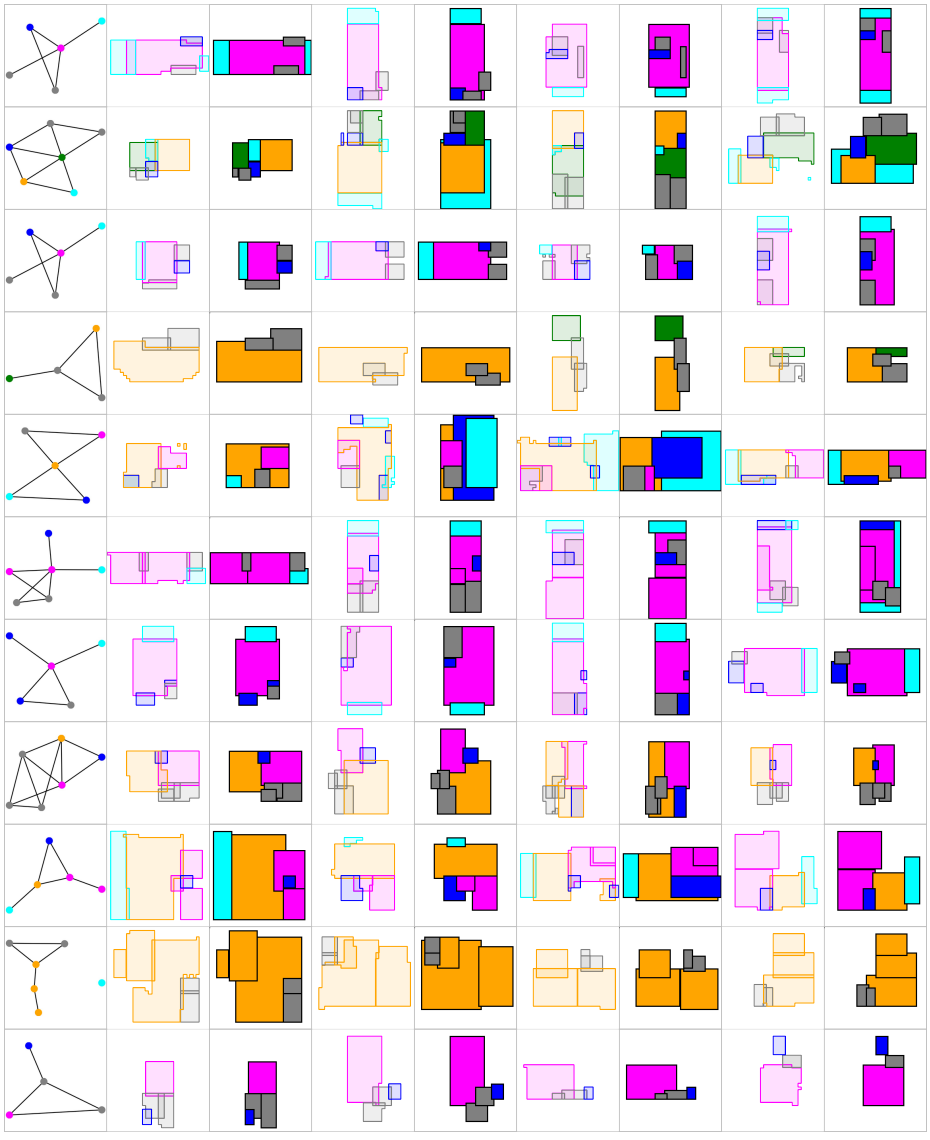

Fig. 5. Continued.

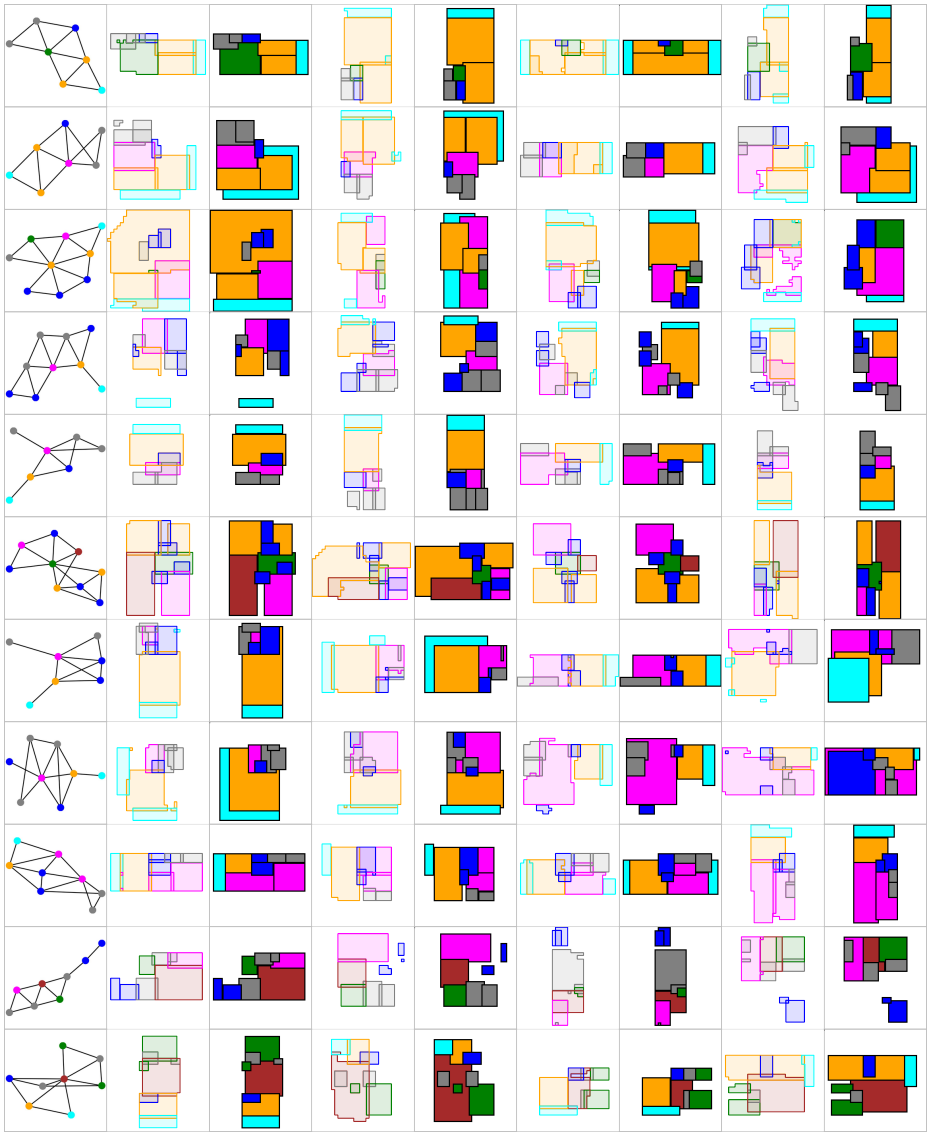

Fig. 6. Results for model trained on graphs of size 1-6 and 10+ and tested on graphs of size 7-9.

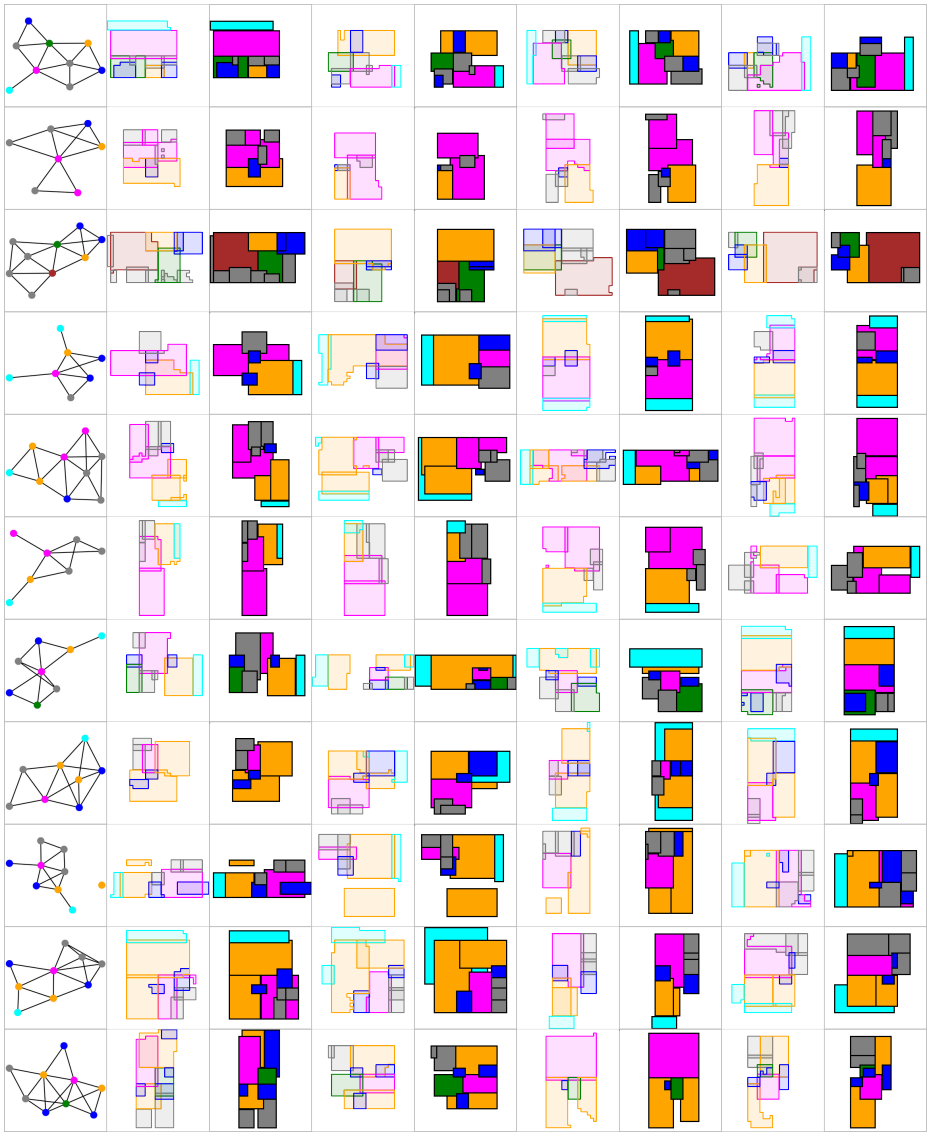

Fig. 7. Continued.

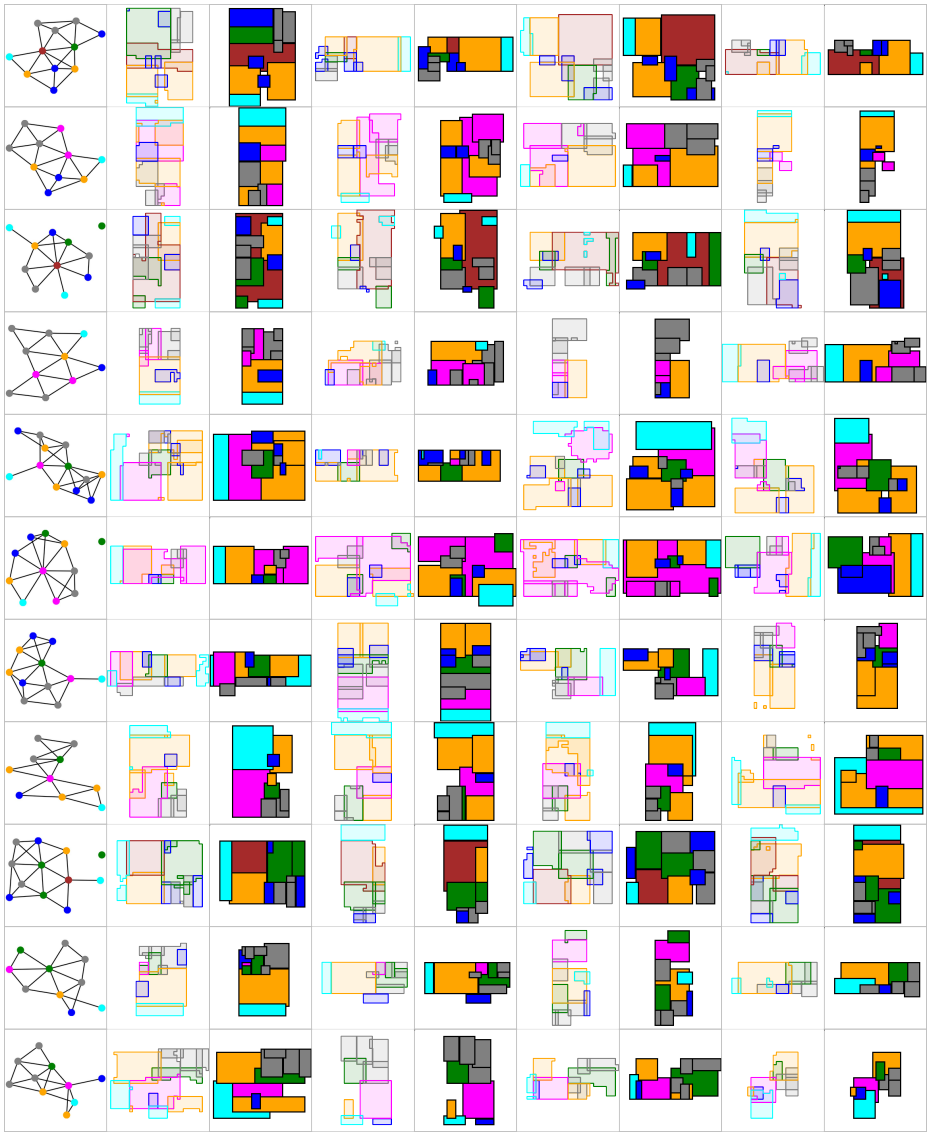

Fig. 8. Results for model trained on graphs of size 1-9 and 13+ and tested on graphs of size 10-12.

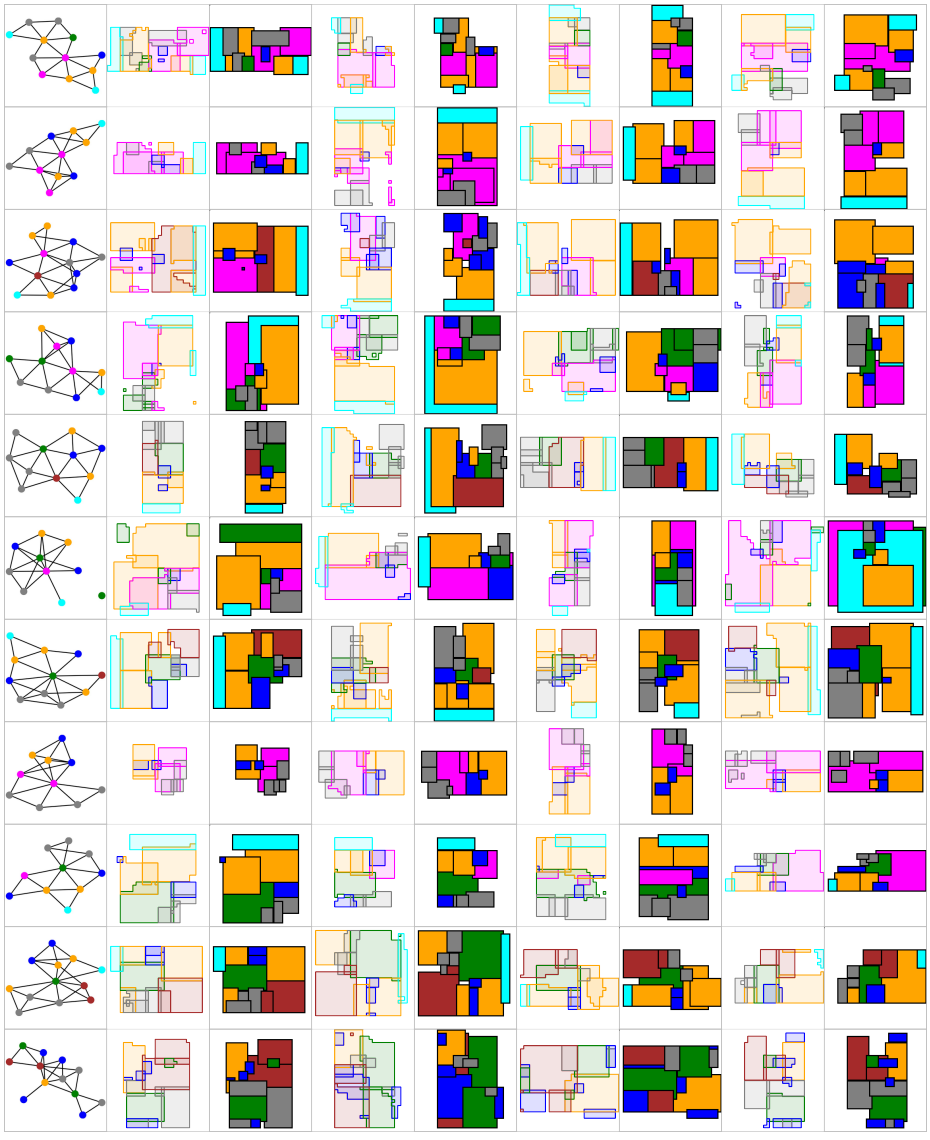

Fig. 9. Continued.

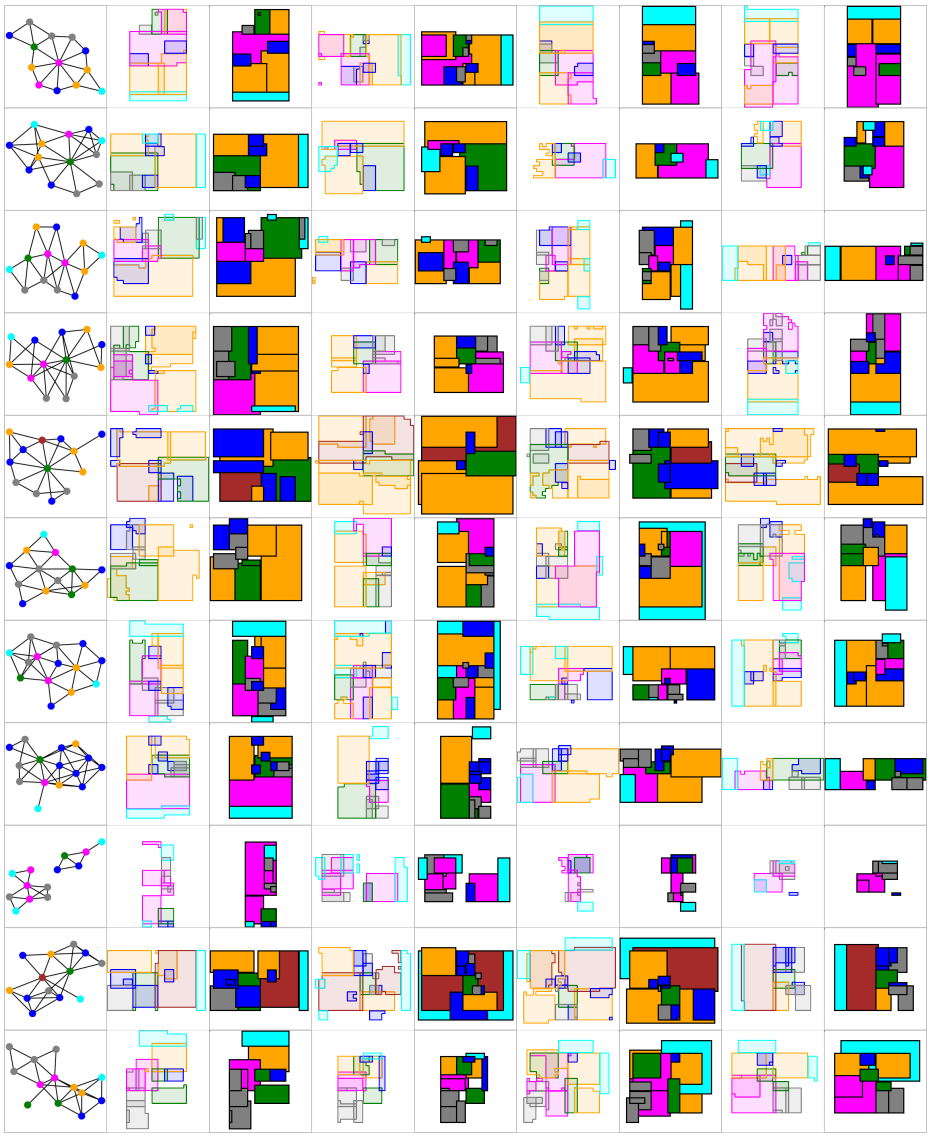

Fig. 10. Results for model trained on graphs of size 1-12 and tested on graphs of size 13+.

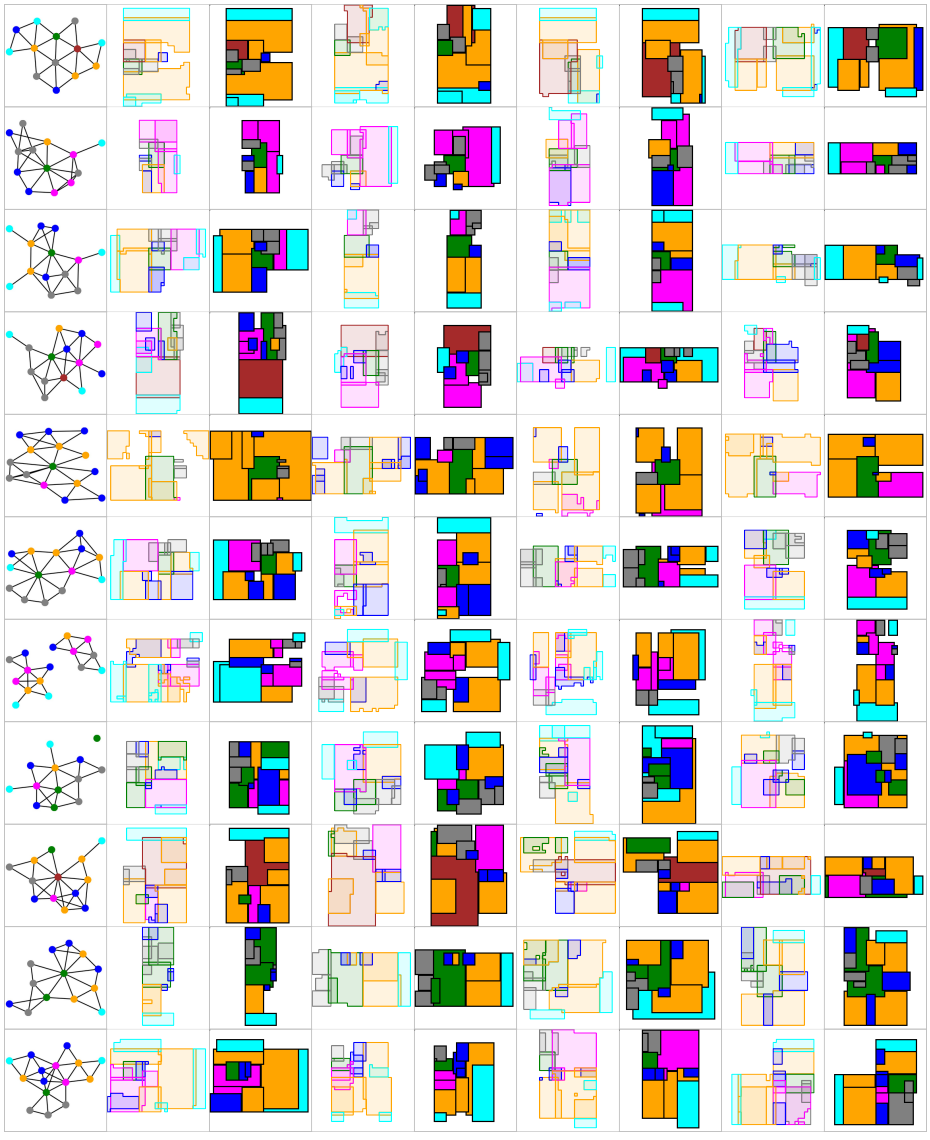

Fig. 11. Continued.

References

1. Ashual, O., Wolf, L.: Specifying object attributes and relations in interactive scene generation. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4561–4569 (2019)
2. Hagberg, A., Swart, P., S Chult, D.: Exploring network structure, dynamics, and function using networkx. Tech. rep., Los Alamos National Lab.(LANL), Los Alamos, NM (United States) (2008)
3. Johnson, J., Gupta, A., Fei-Fei, L.: Image generation from scene graphs. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1219–1228 (2018)