Order Matters: Learning Element Ordering for Graphic Design Generation
Supplementary Material

BO YANG, ShanghaiTech University, China
YING CAO*, ShanghaiTech University, China

1 DATASET PREPROCESSING

1.1 Crello dataset

We filter out designs that can not be rendered properly by our rendering method due to opacity of elements, rotation, and missing CSS styles. First, since our design sequence representation does not include alpha attributes, elements with transparency in the original design may be rendered as fully opaque (alpha=1.0), causing them to completely obscure underlying elements. Thus, we remove all samples containing elements with alpha values less than 0.95 or mask elements, whose proper display requires alpha channel information. Second, since we do not consider the rotation attributes of elements, we filter out samples containing elements with rotation greater than 45 degrees, which can not be rendered well without the rotation attributes. Third, some elements have very different renderings from their original appearance, due to missing CSS files. To handle these cases, we calculate the structural similarity index measure (*SSIM*) [Wang et al. 2004] between rendered designs and their actual images, and filter out samples with SSIM less than 0.8.

We further filter out samples containing an excessive number of tiny elements. Specifically, we define tiny elements as those occupying less than 0.01% of the canvas and discard samples where tiny elements constitute 50% or more of the total elements. We end up with 10,162 design samples with an average of 9.35 elements per design. We show the distributions of element counts per sample for each element category in the filtered Crello dataset in Figure 1.

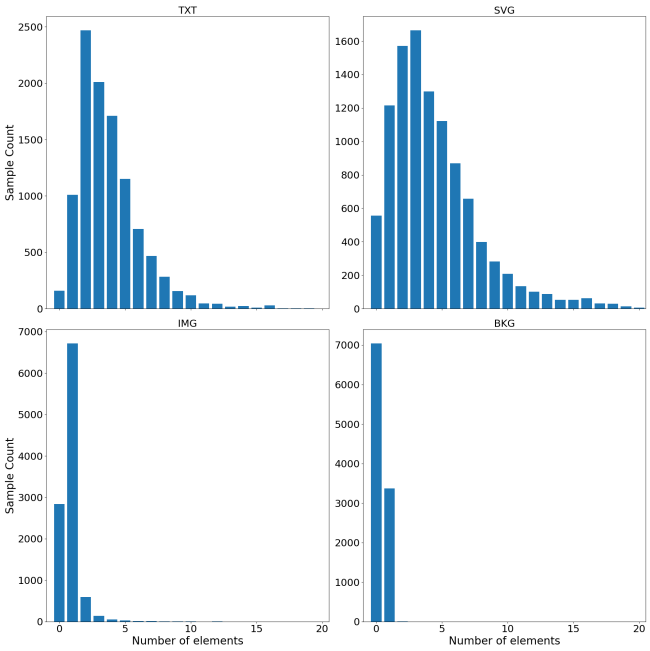

Fig. 1. Distributions of element counts per sample in the filtered Crello dataset for each of the four element categories: text (TXT), scalable vector graphics (SVG), image (IMG), background (BKG).

1.2 CLAY dataset

We first remove design samples containing invalid elements based on CLAY's annotations, and retain only designs with no more than 20 elements. This gives rise to 25,540 samples. We then apply the following three filtering operations to the obtained samples:

- Offset-based filtering. To handle annotation inconsistency on CLAY, where overall annotation has significant misalignment with the UI screenshot (Figure 2 (left)), we compute the minimum enclosing rectangle for all elements, and measure its horizontal center offset from the canvas center. Samples with the offsets larger than 0.15 are removed.

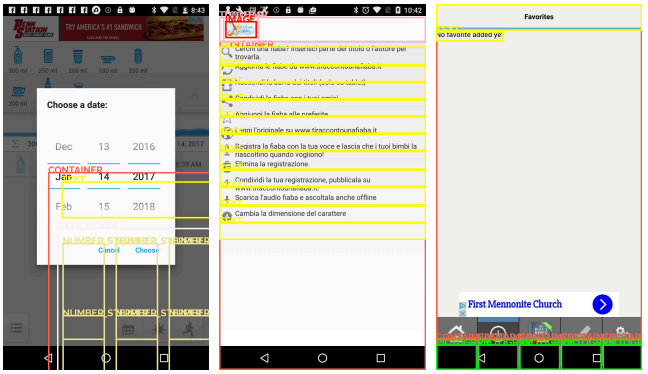

Fig. 2. Some typical design samples to be filtered out in the CLAY dataset.

*Corresponding author.

Authors' addresses: Bo Yang, yangbo2022@shanghaitech.edu.cn, ShanghaiTech University, Shanghai, China; Ying Cao, yingcao59@gmail.com, ShanghaiTech University, Shanghai, China.

ACM 0730-0301/2025/8-ART
https://doi.org/10.1145/3730858

- Alignment-based filtering. To encourage richer design variations in samples, we identify and remove designs with highly regular layouts. We quantify layout regularity using alignment between elements, since more regular layouts tend to have better alignment (Figure 2 (middle)). we use two alignment metrics: *Alignment-HV* [Kikuchi et al. 2021] that considers both horizontal and vertical alignment between elements and *Alignment-H* [Lee et al. 2020] that specifically focuses on horizontal alignment. Samples are retained only if their Alignment-HV scores are larger than 0 and their Alignment-H scores are larger than 0.0001. The assignment scores are calculated based on normalized element bounding box coordinates.
- Coverage-based filtering. We further discard mobile UI designs with too small negative space (Figure 2 (right)), by removing samples where all components cover more than 95% of the canvas.

After applying the above filtering operations, we obtain 19,631 samples.

2 DESIGN ATTRIBUTE QUANTIZATION DETAILS

We perform quantization on various design attributes (Section 3.1 of the main paper) as follows. On all the three datasets (Crello, CGL, CLAY), we quantize each of bounding box coordinates (x, y, w, h) into 128 bins. For font size and color (f_s, f_c) that are provided in Crello, we quantize each of them into 128 bins.

For each subcategory c of non-text elements, we perform quantization on the image attribute u. Table 1 shows the number of bins for each subcategory for different datasets. We adaptively adjust the bin counts for some subcategories according to their frequencies. For consistency, we treat the background of each CGL sample as a distinct element with a "background" subcategory and perform quantization on its image attribute as well. For training the diffusion model (Section 5.2.3 of the main paper), we reduce the number of quantization bins to 512 for all the subcategories of non-text elements in Crello due to memory constraints.

For canvas attributes, we uniformly quantize the canvas color r into 32 bins across all the datasets. For Crello, canvas width and height (w, h) are not quantized and we directly use 38 possible values for w and 41 possible values for h that occur in Crello. For CLAY, both w and h are quantized into 32 bins; for CGL, only a single canvas size is used.

Table 1. Number of quantization bins for each subcategory of non-text elements in different datasets.

Dataset	Subcategory	Bins
Crello	SVG	2048
	Image	2048
	Colored Background	512
CGL	Logo	1024
	Text	1024
	Underlay	1024
	Embellishment	1024
	Background	1024
CLAY	Background, Image, Pictogram	1024
	Button, Text, Text Input, Label	1024
	List Item, Container, Card View	1024
	Navigation Bar, Toolbar, Page Indicator	1024
	Check Box, Switch, Spinner	512
	Drawer	512
	Radio Button, Progress Bar	256
	Map, Slider	128
	Date Picker, Number Stepper	128
	Advertisement	64

3 LAYER DEPTH ESTIMATION

One of the baselines considered in the paper, saliency order, uses the layer depth of elements to determine occlusion relationships between elements for more accurate element-wise importance score computation. Another baseline, layer order, heavily relies on layer depth for sorting elements. For layered graphic design dateset, e.g., Crello, the layer depth for each element is given directly. For CGL and CLAY where such depth information is not available, we approximate it based on some simple heuristic rules.

For CGL, the layer depth of elements is estimated according to their categories, following layering convention in graphic design. In particular, the layer hierarchy from bottom to top is defined as: background → underlay → embellishment → text → logo. This ordering approach reflects common design practice, where logos and text typically appear above decorative elements and the background.

For CLAY, layer depth estimation is based on both the categories and sizes of elements. First, we make the group of text-related elements (text, text input, and label) appear above the group of non-text elements. Within each group, we then determine depth based on element sizes: smaller elements are assigned smaller depth values compared to larger elements. This aligns with common UI design patterns, where smaller, interactive elements are typically placed above larger container elements.

4 USER STUDY DETAILS

To evaluate the perceptual quality of generated designs, we run a user study involving 28 participants, including both novice and expert designers. The study was conducted through a web-based platform. Participants were tasked with comparing given designs and selecting their favorite ones based on several design aspects: (1) layout & composition; (2) visual coherence; (3) typography & readability. These evaluation criteria and their detailed explanations, along with the task instruction, were presented to the participants at the beginning of the study, as shown in Figure 3.

Each participant was asked to complete 15 rounds of evaluation. As shown in Figure 4, in each round, participants were shown six groups of designs, and asked to choose their favorite design group. Each group contains three designs generated using a specific ordering approach (random, raster, saliency, layer, layer-and-raster or our neural order). To prevent potential position bias, the presentation order of groups was randomized in each round. All the designs were presented at the same canvas size, and the participants could access the designs at full size for detailed examination. No time constraints

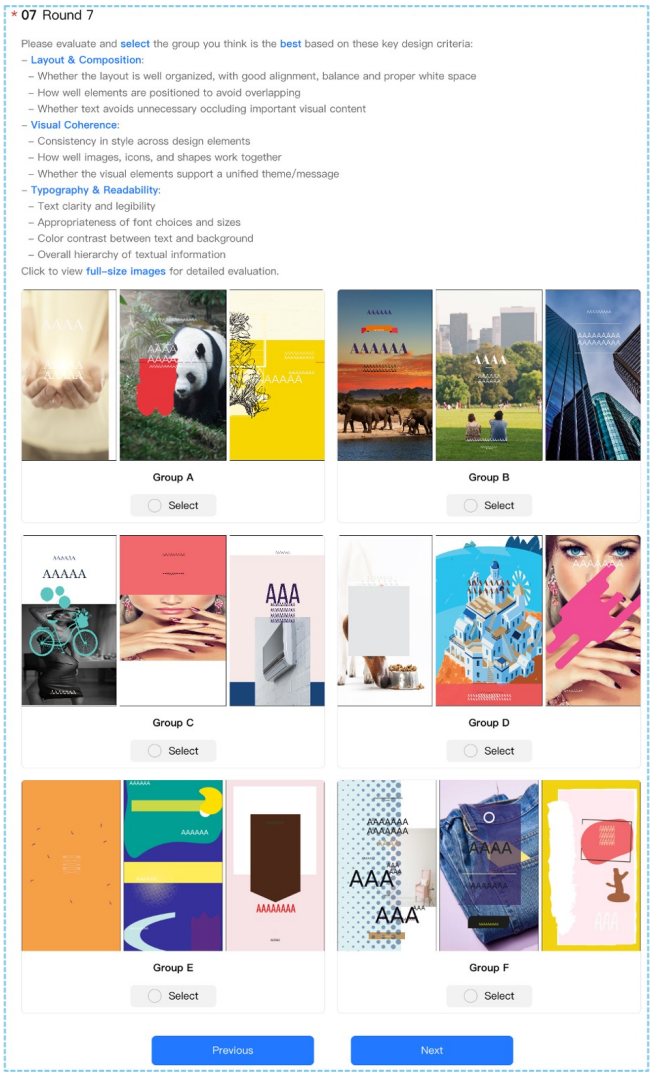

Fig. 4. Example of one evaluation round in the user study.

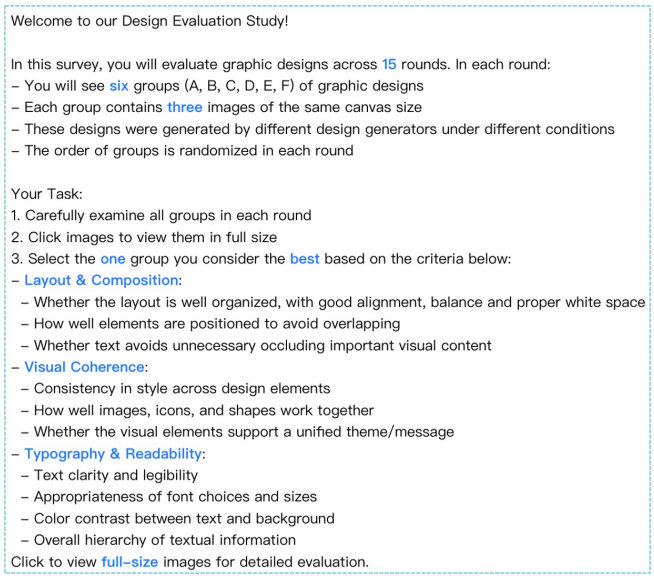

Fig. 3. Task instruction and evaluation criteria provided to participants at the beginning of the study.

were imposed during the study, and the average completion time of one round was about 7 minutes.

5 MORE QUALITATIVE RESULTS

Figure 5 shows the visual comparison of samples generated from autoregressive design generators trained with layer order, layer-and-raster order and our neural order. The results from layer order suffer from several obvious limitations: first, they are overly simple, with few elements (e.g., row 4, 5, 8); second, they predominantly use black and white font colors across all the designs, and sometimes lack sufficient color contrast between text and the background, resulting in poor readability (e.g., row 5, 7); third, they often fail to well align elements, particularly text elements (e.g., row 2, 6, 7). Layer-and-raster order improves overall sample quality upon layer order, but still exhibits some issues, such as simplistic designs (e.g., row 1, 3, 5), occlusion of important image contents (e.g., row 4, 7, and 8), and undesirable overlap between elements (e.g., row 6, 7). In contrast, our neural order is considerably better than the other two orders in sample quality. Figure 6 shows additional results sampled from the autoregressive design generator trained with our neural order.

REFERENCES

Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. 2021. Constrained Graphic Layout Generation via Latent Optimization. In *Proceedings of the 29th ACM International Conference on Multimedia*. 88–96.

Hsin-Ying Lee, Lu Jiang, Irfan Essa, Phuong B. Le, Haifeng Gong, Ming-Hsuan Yang, and Weilong Yang. 2020. Neural Design Network: Graphic Layout Generation with Constraints. In *Proceedings of the 16th European Conference on Computer Vision*. 491–506.

Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. 2004. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing* 13, 4 (2004), 600–612.

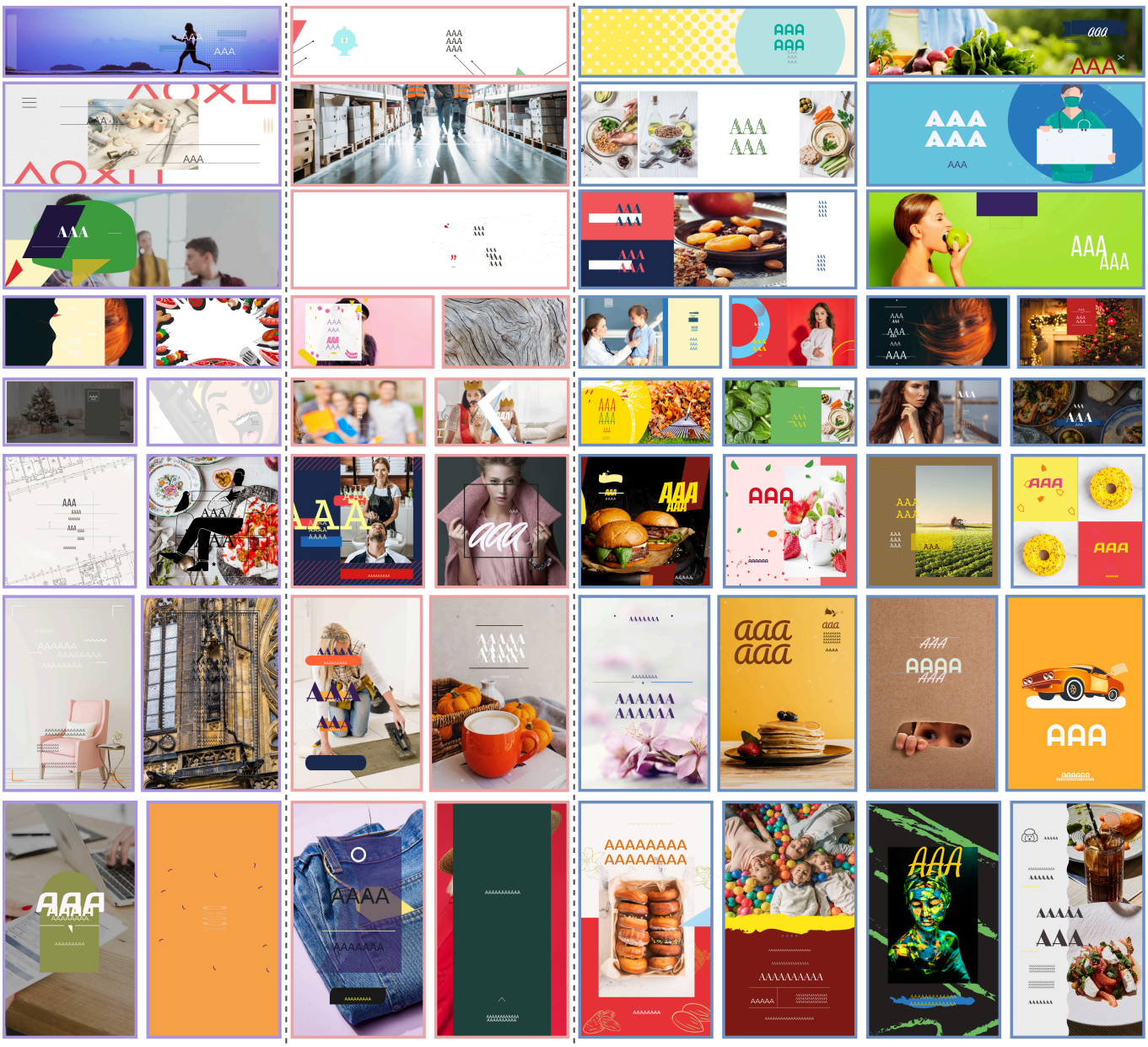

Fig. 5. Qualitative comparison of samples generated from models trained with different orders.

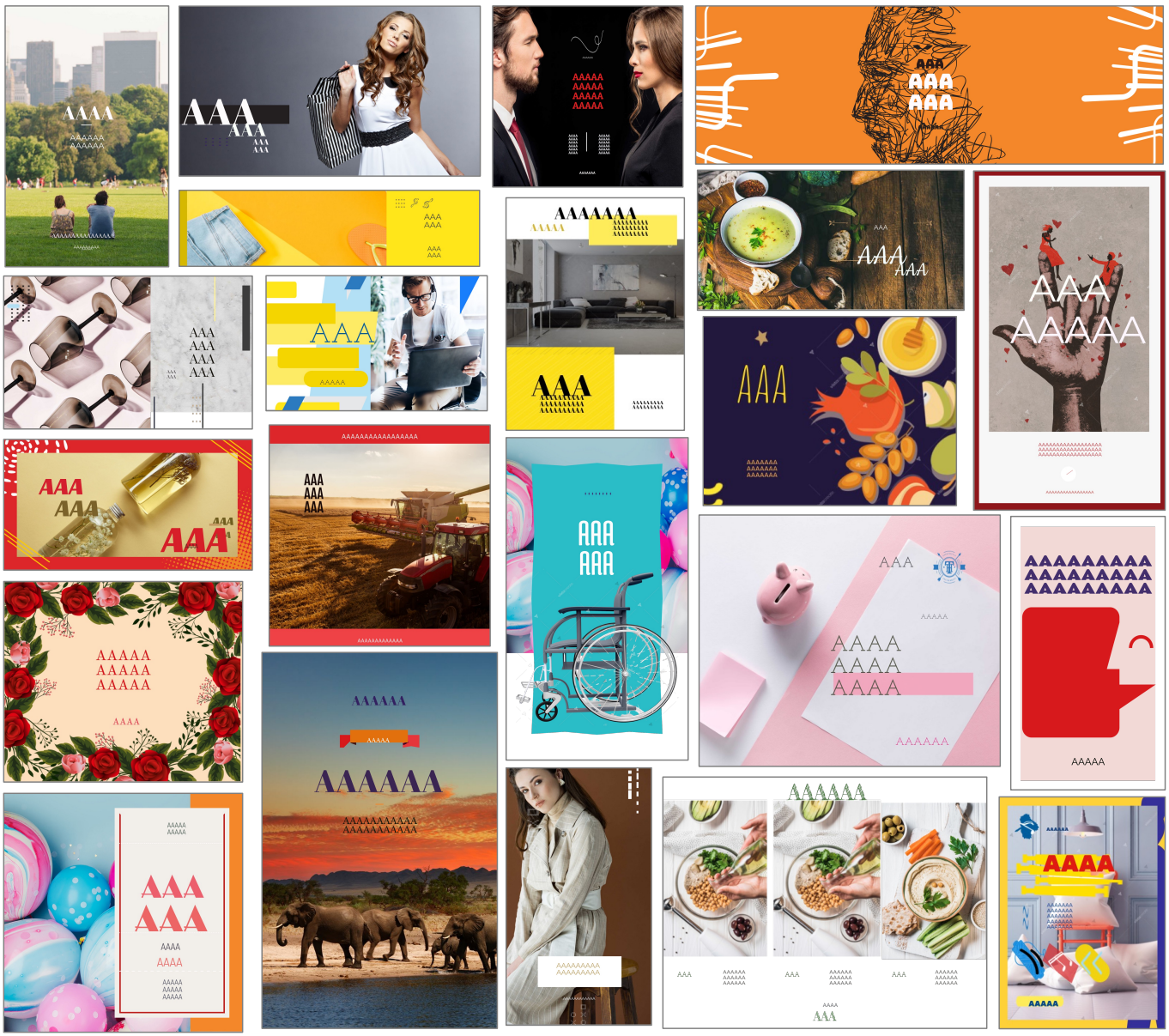

Fig. 6. Samples generated from the autoregressive design generator trained with our neural order.