Supplemental Materials

A. Details of typographic attributes

Font: We categorize fonts based on font family name. We do not model the similarity of fonts in this paper.
Color: The color attribute is the color of the texts for filling.
Alignment: If texts have line breaks, the alignment attribute aligns texts in left, right, or center.
Capitalization: The capitalization attribute is a binary option whether to capitalize texts or not.
Font size: The font size attribute is an input parameter of font, and it controls the size of texts.
Angle: The angle of texts for rotation.
Letter spacing: The letter spacing attribute represents the distance of letters in texts.
Line spacing: The line spacing attribute is a scale of interval in lines.

B. Dataset statistics

We show statistics of typographic attributes of the Crello dataset [3]. Fig. 10 shows the distribution. We observe strong biases in typographic attributes that designers prefer to use.

Fig. 11 shows the number of unique labels in typographic attributes per design in the Crello dataset. Even if there are many text elements, there are only a few attributes in use, and we rarely observe more than three different fonts in a single design document. Geometric attributes like font size or line spacing tend to have fewer counts than semantic attributes like font or color. Note that we show the discretized label count for geometric attributes and color.

C. Architecture details

We describe the details of our encoder-decoder architecture in the following.

Encoder For each input feature, we project the feature x_i into an embedding $\mathbf{z}_i$ using an encoder: $\mathbf{z}_i = E_i(x_i; \theta)$. We apply the same encoder to all of the element contexts, where i is an index to the input modalities and elements; i.e., $i = (k, t)$ indicates the k-th attribute of the t-th element. For the image feature, we apply ImageNet pre-trained ResNet50 [1] to obtain a feature representation. We apply a pre-trained CLIP [2] to encode a text input. For other categorical features, we apply one-hot encoding to obtain a vector representation. Once we obtain modality-wise features, we apply a linear projection to all of the features, concatenate all of them into a sequence, and obtain fixed-dimensional embeddings $Z \equiv \{\mathbf{z}_i\}$. Let us also denote the set of embeddings belonging to the t-th element by $\mathbf{z}_t$ and to the canvas by $\mathbf{z}_{\text{canvas}}$.

We further apply self-attention transformer modules F to the latent sequence: $Z' \equiv \{\mathbf{z}'_i\} = F_{\text{encoder}}(Z; \theta)$ so that the attention mechanism captures any interaction between different modalities across text elements.

Decoder We adopt an autoregressive decoder to model the joint distribution of typographic attributes:

$$p_\theta(Y|X) = \prod_t^T p_\theta(\mathbf{y}_t|\mathbf{y}_{t-1}, \dots, \mathbf{y}_1, X), \tag{1}$$

and we apply element-wise autoregressive sampling to generate attribute k at the t-th element:

$$\hat{y}_k^t \sim p_\theta(y_k^t|\mathbf{y}_{t-1}, \dots, \mathbf{y}_1, X). \tag{2}$$

We build the decoder architecture in the following approach:

$$p_\theta(y_k^t|\mathbf{y}_{t-1}, \dots, \mathbf{y}_1, X) \equiv F_k(\mathbf{h}'_t, \mathbf{s}_t; \theta), \tag{3}$$
$$\mathbf{h}'_t = F_{\text{decoder}}(Z', H_t; \theta), \tag{4}$$
$$\mathbf{s}_t = F_{\text{skip}}(\mathbf{z}_t, \mathbf{z}_{\text{canvas}}; \theta). \tag{5}$$

We model the categorical distribution of each attribute k by the softmax function in the decoder head F_k. Our decoder head takes concatenated features with the outputs from the decoder Transformer F_{decoder} and the skip connection $F_{\text{skip}}(\mathbf{z}_t, \mathbf{z}_{\text{canvas}})$ which is a shallow MLP. Our decoder Transformer takes the latent sequence Z' from the encoder and the query sequence $H_t \equiv \{\mathbf{h}_1, \dots, \mathbf{h}_t\}$ where:

$$\mathbf{h}_t \equiv \mathbf{p}_t + \sum_{k \in \mathcal{K}} E_k(y_k^{t-1}), \tag{6}$$

which is a sum of the positional encoding $\mathbf{p}_t$ and additive pooling of the attribute embeddings for $\mathbf{y}_{t-1}$ at the t-th text element. We use the raster scan order of elements, i.e., from

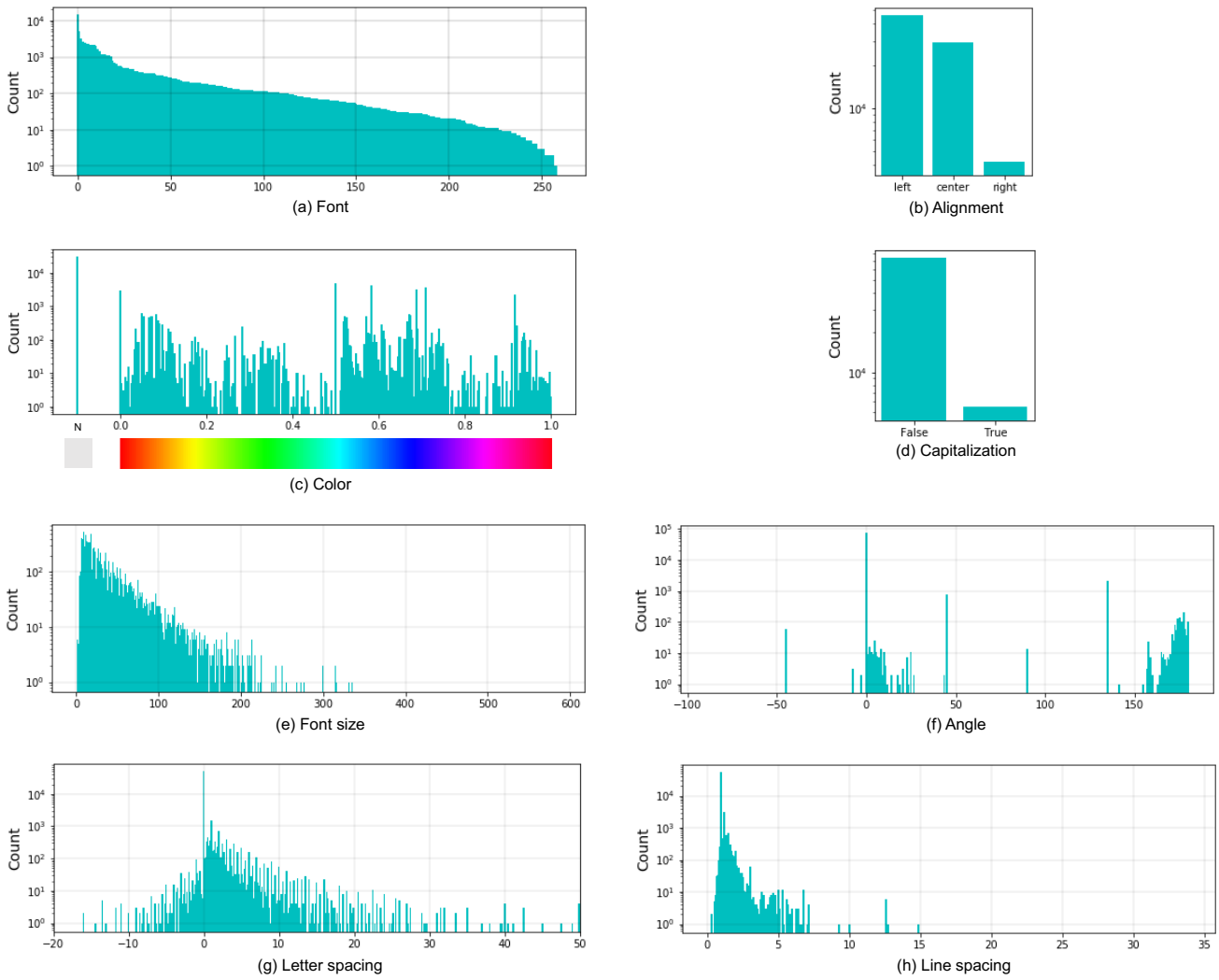

Figure 10. The distributions of typographic attributes in the Crello dataset. The y-axis of plots is the logarithmic scale. N in the color distribution represents the neutral color. The (a) to (d) are semantic attributes, and (e) to (h) are geometric attributes.

Table 4. Attribute metrics. Acc: accuracy, MAE: mean absolute error, and Diff: CIEDE2000 color difference.

TF	Skip	Font Acc (%) ↑	Color Diff (-) ↓	Alignment Acc (%) ↑	Capitalization Acc (%) ↑	Font size MAE (pt) ↓	Angle MAE (°) ↓	Line spacing MAE (pt) ↓	Line height MAE (-) ↓
	✓	35.3±0.76	53.3±1.69	92.8±0.68	73.5±1.02	23.0±3.34	0.30±0.06	2.15±0.14	0.065±0.001
✓		39.5±0.48	54.0±0.96	93.2±0.64	72.5±0.75	27.7±0.84	0.33±0.05	2.26±0.15	0.092±0.006
✓	✓	40.9±0.76	53.7±1.96	93.8±0.74	75.3±0.67	20.9±0.66	0.26±0.02	2.16±0.16	0.065±0.003

top-left to bottom-right, to represent the order of the elements. $\mathcal{K}$ is a set of typographic attributes for each element. For $t = 1$, we prepare a special `[start]` token for the second term. We use the raster scan order, i.e., from top-left to bottom-right, to define the order of elements.

D. Architecture ablation

We ablate the architecture of our model in this section. We verify the effectiveness of two components the transformer blocks "TF" and the skip connection "Skip". Tables 4 and 5 summarize the prediction performance. We observe that the features from the Transformer blocks improve the prediction of the font and alignment attributes. While they degrade the performance of prediction in other attributes from the shallow features, i.e., the features from skip-connection, the combined features improve the performance. These results indicate that both deep features from transformer blocks and shallow features improve the prediction of typographic attributes. In terms of struc-

Table 5. Structure scores (%).

TF	Skip	Font	Color	Alignment	Capitalization	Font size	Angle	Line spacing	Line height
	✓	54.3±0.66	60.1±0.71	64.3±0.81	84.2±0.78	68.0±0.61	84.8±1.16	61.3±1.18	79.1±0.75
✓		67.2±0.68	65.2±0.36	66.0±0.65	86.1±0.66	67.5±0.58	84.1±1.12	62.3±1.29	70.6±0.49
✓	✓	68.6±0.44	66.9±0.65	68.1±0.58	86.3±0.55	71.3±0.55	86.0±0.37	63.8±0.77	78.9±1.06

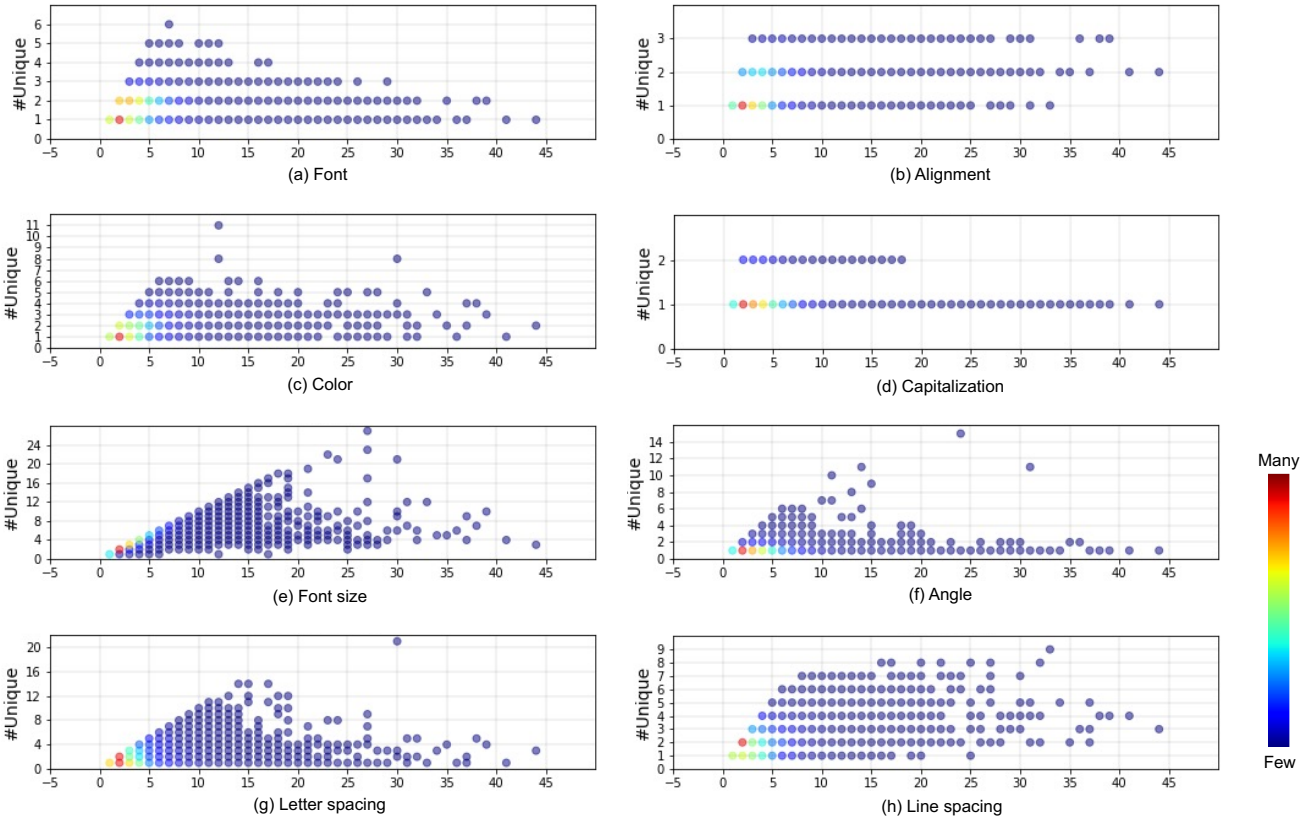

Figure 11. The use of typographic attributes by the number of text elements. The color represents the count of unique labels.

ture score, the prediction performance through Transformer blocks shows better scores compared to the shallow features in non-geometric attributes and line spacing. Also, combined features consistently improve the performance except for line spacing.

E. Additinal qualitative results

Fig. 12 shows additional generation examples. Our model successfully generates appropriate typography in various situations, e.g., many text elements, small text, and large text. We also show the generated examples with different hyper-parameters p in Fig. 13. The sensitivity of hyper-parameters depends on the context.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[2] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. *arXiv preprint arXiv:2103.00020*, 2021. 1

[3] Kota Yamaguchi. CanvasVAE: Learning to generate vector graphic documents. In *IEEE International Conference on Computer Vision (ICCV)*, 2021. 1

Figure 12. Additional diverse generation examples. Each row shows three generated examples for the same input.

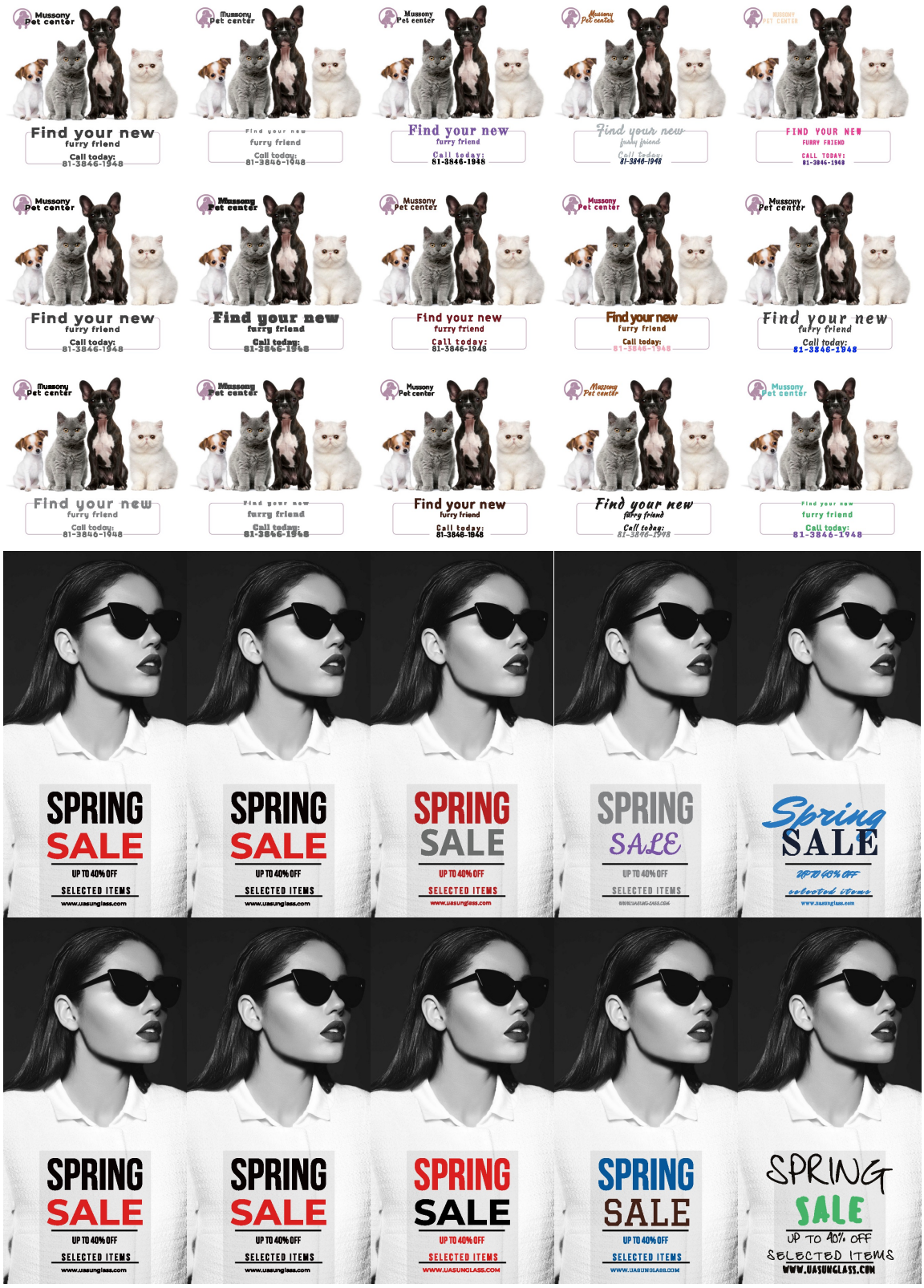

Figure 13. Generated examples with different diversity hyper-parameter p.