Retrieval-Augmented Layout Transformer for Content-Aware Layout Generation

Daichi Horita[1] Naoto Inoue[2] Kotaro Kikuchi[2] Kota Yamaguchi[2] Kiyoharu Aizawa[1]
[1]The University of Tokyo [2]CyberAgent
{horita,aizawa}@hal.t.u-tokyo.ac.jp
{inoue_naoto,kikuchi_kotaro_xa,yamaguchi_kota}@cyberagent.co.jp

Abstract

Content-aware graphic layout generation aims to automatically arrange visual elements along with a given content, such as an e-commerce product image. In this paper, we argue that the current layout generation approaches suffer from the limited training data for the high-dimensional layout structure. We show that a simple retrieval augmentation can significantly improve the generation quality. Our model, which is named Retrieval-Augmented Layout Transformer (RALF), retrieves nearest neighbor layout examples based on an input image and feeds these results into an autoregressive generator. Our model can apply retrieval augmentation to various controllable generation tasks and yield high-quality layouts within a unified architecture. Our extensive experiments show that RALF successfully generates content-aware layouts in both constrained and unconstrained settings and significantly outperforms the baselines. [1]

1. Introduction

Layout is an essential part of graphic design, where the aesthetic appeal relies on the harmonious arrangement and selection of visual elements such as logos and texts. In real-world creative workflows, such as posters [13, 33] and magazines [20, 44] creation, designers typically work on a given subject; for example, creating an advertising poster of a specific product. We call layout generation under such conditions *content-aware* layout generation, where the goal is to generate diverse yet plausible arrangements of element bounding boxes that harmonize with the given background image (canvas). Recent studies [47, 48] show that generative models can produce content-aware layouts that respect aesthetic principles, such as avoiding overlaps [13]. However, generated layouts often still suffer from artifacts, including misaligned underlay embellishment and text elements. We hypothesize that current approaches based solely on generative models do not scale due to the scarcity of highly structured layout data. Unlike public images on the Web, curating a large dataset of layered graphic designs is not a viable solution since designers typically create their work in proprietary authoring tools, such as Adobe Illustrator [1].

Figure 1. Retrieval-augmented content-aware layout generation. We retrieve nearest neighbor examples based on the input image and use them as a reference to augment the generation process.

Inspired by the fact that designers often refer to existing designs [17], we propose a retrieval-augmented generation method to address the challenges in the layout domain. Recent literature shows that retrieval augmentation helps in enhancing the generation quality of language models [6, 15] and image synthesis [5, 40], thanks to the ability to reference real examples in the limited data domain. We argue that retrieval augmentation plays an important role in mitigating the data scarcity problem in content-aware layout generation.

We build **R**etrieval-**A**ugmented **L**ayout Trans**F**ormer (RALF), which is an autoregressive generator capable of referencing external layout examples. RALF retrieves reference layouts by nearest neighbor search based on the appearance of the input and supplements the generation process (Fig. 1). Since the input canvas and retrieved layouts have different modalities, we use the cross-attention mechanism to augment the feature input to the generator. Although we build RALF with an autoregressive approach, retrieval augmentation is also effective in other generation approaches such as diffusion models [19], which we show in the experiments.

We evaluate our RALF on public benchmarks [18, 48] and show that RALF outperforms state-of-the-art models in content-aware layout generation. Thanks to the retrieval capability, RALF requires less than half the training data to achieve the same performance as the baseline. We further show that our modular architecture can adapt to *control-*

[1]Our project page is available at https://udonda.github.io/RALF/

lable generation tasks that impose various user-specified constraints, which is common in real-world workflow.

We summarize our contributions as follows: 1) We find that retrieval augmentation effectively addresses the data scarcity problem in content-aware layout generation. 2) We propose a Retrieval-Augmented Layout Transformer (RALF) designed to integrate retrieval augmentation for layout generation tasks. 3) Our extensive evaluations show that our RALF successfully generates high-quality layouts under various scenarios and significantly outperforms baselines. We will make our code publicly available on acceptance.

2. Related Work

2.1. Content-agnostic Layout Generation

Content-agnostic layout generation, which aims at generating layouts without a specific input canvas, has been studied for a long time [2, 32, 33, 44]. The typical approach involves predicting the arrangement of elements, where each element has a tuple of attributes such as category, position, and size [29]. Recent approaches employ various types of neural networks-based generative models, such as generative adversarial networks (GAN) [24, 29, 30], variational autoencoders (VAE) [3, 21, 23], autoregressive models [14, 22], non-autoregressive models [25], and diffusion models [9, 19, 27, 45]. Note that the retrieval augmentation discussed in this paper may not be directly applicable to the content-agnostic setup due to the lack of input queries.

Several works consider user-specified design constraints such as "a title is above the body", which are often seen in real-world workflow. Such constraints are studied as controllable generation [19, 22, 24, 25], where the model generates a complete layout from a partial or noisy layout. In this paper, we adapt the concept of controllable generation to the content-aware generation.

2.2. Content-aware Layout Generation

Content-aware layout generation, relatively less studied compared to the content-agnostic setup, has seen notable progress. ContentGAN [47] first tackles to incorporate image semantics of input canvases. Subsequently, CGL-GAN [48] introduces a saliency map to a non-autoregressive decoder [8, 10, 42] for better subject representation. DS-GAN [18] proposes a CNN-LSTM framework. ICVT [7] employs a conditional VAE, predicting a category and bounding box autoregressively based on previously predicted elements. RADM [28] leverages a diffusion model and introduces modules to refine both visual–textual and textual–textual presentations. We note that we cannot compare RADM in our experiments because their text annotations are not available.

Current approaches rely solely on generative models and may struggle with capturing sparse data distributions with limited training data. We use retrieval augmentation to mitigate this issue, and our experiments confirm its significant impact on enhancing content-aware generation.

2.3. Retrieval-Augmented Generation

Retrieval augmentation [4–6, 15, 40] offers an orthogonal approach to enhance generative models without increasing network parameters or relying heavily on extensive training datasets. Generative models equipped with retrieval augmentation stop storing all relevant knowledge in their model parameters and instead use external memory via retrieving relevant information as needed. A common approach involves retrieving the k-nearest neighbors (k-NN) based on a pre-calculated embedding space as additional input. For example, REALM [15] introduces a retrieval augmentation into language models that fetch k-NN based on preceding tokens. In image generation, RDM [5] demonstrates even a relatively compact network can achieve state-of-the-art performance by retrieval augmentation. KNN-Diffusion [40] shows its capacity to generate out-of-distribution images. The unique challenge in content-aware layout generation involves encoding both image and layout modalities, which we address using a cross-attention mechanism.

Given that tasks related to graphic design, such as content-aware layout generation, often suffer from data scarcity problems [34], we believe that retrieval augmentation is particularly beneficial. It provides an efficient training method that leverages existing data more effectively.

3. Method

3.1. Preliminaries

Let $\mathcal{X}$ and $\mathcal{Y}$ be the sets of canvas images and graphic layouts, respectively. We use $I \in \mathcal{X}$ and $L \in \mathcal{Y}$ to represent the canvas and layout, respectively. The canvas $I \in \mathbb{R}^{H\times W\times 3}$ and layout L are paired data, where H and W represent the height and width, respectively. We obtain a saliency map $S \in \mathbb{R}^{H\times W\times 1}$ by the off-the-shelf saliency detection method [35, 36] from the canvas. We denote the layout by $L = \{l_1, \ldots, l_T\} = \{(c_1, \boldsymbol{b}_1), \ldots, (c_T, \boldsymbol{b}_T)\}$, where $\boldsymbol{b} \in [0, 1]^4$ indicates the bounding box in normalized coordinates, $c_i \in \{1, \ldots, C\}$ indicates an element category of i-th element, and T indicates the number of elements in L.

3.2. Retrieval-Augmented Layout Transformer

We approach content-aware layout generation by referencing similar examples and generating layout tokens $\hat{Z}$ autoregressively. Following content-agnostic layout generation works [14, 22], we quantize each value in the bounding box of the i-th element $\boldsymbol{b}_i$ and obtain representation $[x_i, y_i, w_i, h_i]^T \in \{1, \ldots B\}^4$, where B denotes the number of bins. Here, x, y, w, and h correspond to the tokens for center coordinates, width, and height of the bounding box. We represent an overall layout as a flattened 1D sequence $Z = (bos, c_1, x_1, y_1, \ldots, w_T, h_T, eos) \in \mathbb{N}^{5T+2}$, where bos and eos are special tokens to denote the start and end of

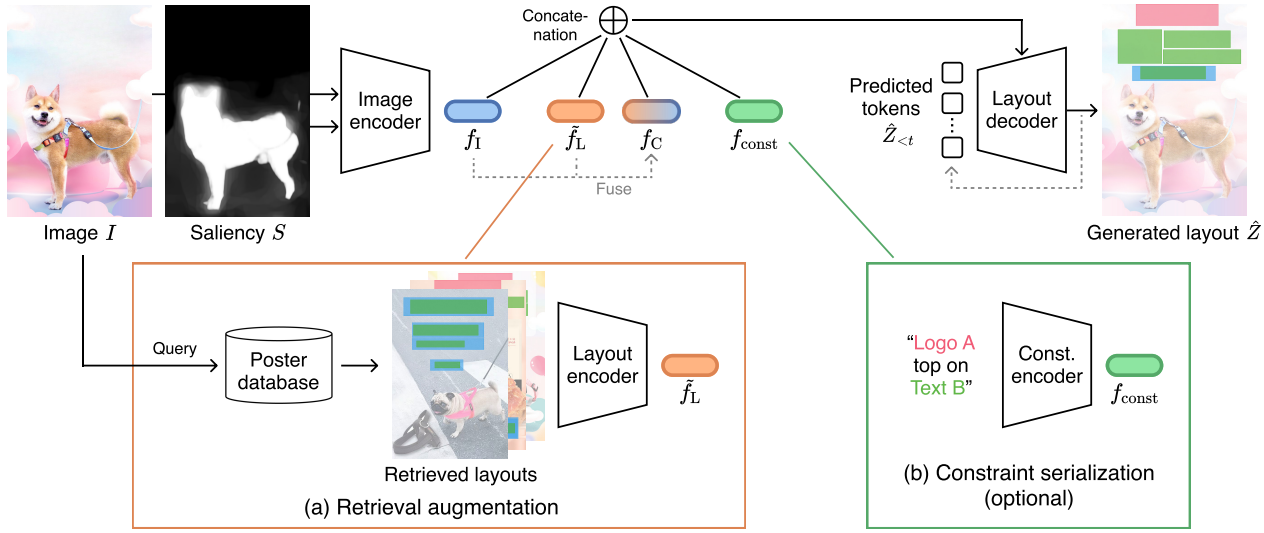

Figure 2. Overview of Retrieval-Augmented Layout Transformer (RALF). RALF takes a canvas image and a saliency map as input, and then autoregressively generates a layout along with the input image. Our model uses (a) retrieval augmentation that incorporates useful examples to better capture the relationship between the image and the layout, and (b) constraint serialization, an optional module that encodes user-specified requirements, enabling the generation of layouts that adhere to specific requirements for controllable generation.

the sequence. We model the joint probability distribution of Z given I and S as a product over a series of conditional distributions using the chain rule:

$$P_\theta(Z|I,S) = \prod_{t=2}^{5T+2} P_\theta(Z_t|Z_{<t},I,S), \tag{1}$$

where θ is the parameters of our model. Similarly to autoregressive language modeling [37], the model is trained to maximize the log-likelihood of the next token prediction.

Our proposed model consists of four modules: image encoder, retrieval augmentation module, layout decoder, and optional constraint encoder, as illustrated in Fig. 2. We describe each module below.

Image encoder. The image encoder E takes in the input canvas I and the saliency map S, and outputs the feature $f_\mathrm{I} = E(I,S) \in \mathbb{R}^{H'W'\times d}$, where H' and W' represent the down-sampled height and width, and d represents the depth of the feature map. This part is common among content-aware approaches, and we follow the architecture of CGL-GAN [48]. The encoder builds on a CNN backbone and a Transformer encoder. The CNN backbone, typically ResNet50 [16], uses a multi-scale feature pyramid network [31]. The Transformer encoder further refines the encoded image feature.

Retrieval augmentation module. The augmentation module transforms the image feature f_I into the augmented feature f_R. We describe the details in Sec. 3.3.

Constraint encoder. Optionally, our model allows control of the layout generation process by additional instruction on desired layout properties such as element types, coordinates, or inter-element relationships. We adopt the Transformer encoder-based model [22] to encode the instructions into a fixed-dimensional vector $f_\mathrm{const} \in \mathbb{R}^{n\times d}$, where n denotes the length of the task-specific sequence. f_const is then concatenated with the augmented feature f_R and fed to the layout decoder.

Layout decoder. Our model autoregressively generates a layout $\hat{Z}$ using a Transformer decoder. Starting from the *bos* token, our decoder iteratively produces output tokens with cross attention to the side feature sequence f_R from the retrieval augmentation module and the optional sequence f_const from the constraint encoder. A key distinction between our model and previous approaches is that we flatten all the attributes into a single sequence for full attention during generation, which is shown effective in content-agnostic layout generation [14, 22]. As we discuss in Eq. (1), we generate layout tokens one by one in $5T+1$ steps using attribute-wise attention. In contrast, GAN-based models [18, 48] generate in one step, and ICVT [7] generates in T steps using element-wise attention.

3.3. Retrieval Augmentation

We introduce retrieval augmentation to effectively learn the structured layout domain with limited training data. The retrieval augmentation module consists of the following three stages: 1) retrieving reference layouts from a database, 2) encoding these layouts into a feature representation, and 3) fusing all features into the final augmented feature f_R. We elaborate on the details of these three stages.

Layout retrieval. Given the input canvas I, we retrieve a set

of useful layout examples $\{\tilde{L}_1, \ldots, \tilde{L}_K\}$, where $K \in \mathbb{N}$. A challenge lies in the absence of joint embedding for image–layout retrieval, unlike the CLIP [38] embedding for image–text retrieval. We hypothesize that given an image–layout pair $(\tilde{I}, \tilde{L})$, $\tilde{L}$ is more likely to be useful when $\tilde{I}$ is similar to I. From a large dataset of image–layout pairs, we retrieve top-K pairs based on image similarity between I and $\tilde{I}$, and extract layouts from these pairs. The choice of the image similarity measure influences the generation quality, as we will discuss in Sec. 4.7 in detail. We use DreamSim [12], which better aligns with human perception of image similarity in diverse aspects such as object appearance, viewing angles, camera poses, and overall layout. All samples from the training split serve as the retrieval source for both training and inference, excluding the query sample from the retrieval source during training to prevent ground-truth leakage.

Encoding retrieved layouts. Each retrieved layout $\{\tilde{L}_1, \ldots, \tilde{L}_K\}$ is encoded into representative features $\tilde{f}_{\mathrm{L}} = \{\tilde{f}_1, \ldots, \tilde{f}_K\} \in \mathbb{R}^{K \times d}$, since each layout has a different number of elements. A layout encoder F embeds each retrieved layout $\tilde{L}_k$ into the representative feature, denoted as $\tilde{f}_k = F(\tilde{L}_k) \in \mathbb{R}^d$. These extracted features are then concatenated into $\tilde{f}_{\mathrm{L}}$. Following [24], we pre-train F in a self-supervised manner and freeze F thereafter.

Feature augmentation. The last step yields the final augmented feature f_{R} by concatenating three features:

$$f_{\mathrm{R}} = \mathrm{Concatenate}(f_{\mathrm{I}}, \tilde{f}_{\mathrm{L}}, f_{\mathrm{C}}) \in \mathbb{R}^{(2H'W'+K) \times d}, \quad (2)$$

where f_{C} is a cross-attended feature between f_{I} and $\tilde{f}_{\mathrm{L}}$:

$$f_{\mathrm{C}} = \mathrm{CrossAttn}(f_{\mathrm{I}}, \tilde{f}_{\mathrm{L}}) \in \mathbb{R}^{H'W' \times d}.$$

In the cross-attention mechanism, the image feature acts as the query, and the retrieved layout feature serves as both the key and value. This design facilitates an interaction between the input canvas and the reference layouts. We then feed the augmented feature f_{R} into the layout generator. We will validate the design of the augmentation module in Sec. 4.7.

4. Experiments

We evaluate our RALF in the unconstrained generation as well as in a variety of constrained generation tasks.

4.1. Datasets

We use two publicly available datasets, CGL [48] and PKU [18], which mainly cover e-commerce posters such as cosmetics and clothing. PKU includes three element categories: *logo*, *text*, and *underlay*, and CGL additionally contains *embellishment* elements. CGL comprises 60,548 annotated posters, *i.e.*, layouts and corresponding images, and 1,000 unannotated canvases, *i.e.*, images only. PKU contains 9,974 annotated posters and 905 unannotated canvases. To obtain canvas–layout pairs for the training, previous works [18, 48] employ image inpainting to remove the visual elements. However, CGL does not provide inpainted posters, and PKU provides inpainted posters with undesirable artifacts. We inpaint the posters of both CGL and PKU using a state-of-the-art inpainting technique [41].

The original datasets do not provide validation and test splits for annotated posters. This limitation prevents fair hyper-parameter tuning, adopting evaluation metrics relying on ground-truth annotations, and the quantitative evaluation of constrained generation tasks since we cannot create constraints from the annotations. To overcome these issues, we create new dataset splits with a train/val/test ratio of roughly 8:1:1. For CGL, we allocate 48,544/6,002/6,002 annotated posters for train/val/test. For PKU, after excluding posters with more than 11 elements and those with elements occupying less than 0.1% of the canvas, we designate 7,735/1,000/1,000 posters for train/val/test. Both datasets have a maximum of 10 elements. For the evaluations, we use the annotated and unannotated test splits.

4.2. Evaluation Metrics

Inspired by the previous works [18, 48], we employ five metrics that evaluate the layout quality both in terms of graphic and content aspects.

Graphic metrics. These metrics evaluate the quality of the generated layouts without considering the canvas. *FID* (↓) for layout [24, 26] has been a primal metric in content-agnostic layout generation, and we adopt this metric in our content-aware scenario. *Underlay effectiveness* (Und ↑) calculates the proportion of valid underlay elements to the total underlay elements. An underlay element is regarded as valid and scores 1 if it entirely covers a non-underlay element; otherwise, it scores 0. *Overlay* (Ove ↓) represents the average Intersection over Union of all element pairs, excluding underlay elements.

Content metrics. These metrics evaluate whether the generated layouts harmonize with the canvas. *Occlusion* (Occ ↓) computes the average saliency value in the overlapping region between the saliency map S and the layout elements. *Readability score* (Rea ↓) evaluates the non-flatness of text elements by calculating gradients in the image space along both vertical and horizontal axes within these elements.

4.3. Baseline Methods

We compare the following methods in the experiments.
CGL-GAN [48] is a non-autoregressive encoder–decoder model employing a Transformer architecture. The model takes in the empty or layout constraint to the decoder.
DS-GAN [18] is a non-autoregressive model using a CNN-LSTM architecture. DS-GAN is only applicable to the unconstrained task because of the internal sorting algorithm.
ICVT [7] is an autoregressive model that combines a Transformer with a conditional VAE.
LayoutDM† [19] is a discrete state-space diffusion model

Method	#Params	PKU					CGL				
		Content		Graphic			Content		Graphic		
		Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓
Real Data	-	0.112	0.0102	0.99	0.0009	1.58	0.125	0.0170	0.98	0.0002	0.79
Top-1 Retrieval	-	0.212	0.0218	0.99	0.002	1.43	0.214	0.0266	0.99	0.0005	0.93
CGL-GAN [48]	41M	0.138	0.0164	0.41	0.074	34.51	0.157	0.0237	0.29	0.161	66.75
DS-GAN [18]	30M	0.142	0.0169	0.63	0.027	11.80	0.141	0.0229	0.45	0.057	41.57
ICVT [7]	50M	0.146	0.0185	0.49	0.318	39.13	**0.124**	0.0205	0.42	0.310	65.34
LayoutDM† [19]	43M	0.150	0.0192	0.41	0.190	27.09	0.127	0.0192	0.82	0.020	2.36
Autoreg Baseline	41M	0.134	0.0164	0.43	0.019	13.59	0.125	0.0190	0.92	0.011	2.89
RALF (Ours)	43M	**0.119**	**0.0128**	**0.92**	**0.008**	**3.45**	0.125	**0.0180**	**0.98**	**0.004**	**1.32**

Table 1. Unconstrained generation results on the PKU and CGL test split. Our RALF outperforms the Autoreg Baseline and achieves the best score on almost all metrics. For reference, we show the Real Data and the Top-1 Retrieval baselines, which do not have a generator.

Method	PKU unannotated				CGL unannotated			
	Content		Graphic		Content		Graphic	
	Occ ↓	Rea ↓	Und ↑	Ove ↓	Occ ↓	Rea ↓	Und ↑	Ove ↓
CGL-GAN	0.191	0.0312	0.32	0.069	0.481	0.0568	0.26	0.269
DS-GAN	0.180	0.0301	0.52	0.026	0.435	0.0563	0.29	0.071
ICVT	0.189	0.0317	0.48	0.292	0.446	0.0425	0.67	0.301
LayoutDM†	0.165	0.0285	0.38	0.201	0.421	0.0506	0.49	0.069
Autoreg Baseline	0.154	0.0274	0.35	0.022	0.384	0.0427	0.76	0.058
RALF (Ours)	**0.133**	**0.0231**	**0.87**	**0.018**	**0.336**	**0.0397**	**0.93**	**0.027**

Table 2. Unconstrained generation results on the PKU and CGL unannotated test split, which is real data without inpainting artifacts.

Method	Retrieval	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓
CGL-GAN		**0.138**	**0.0164**	0.41	0.074	34.51
CGL-GAN	✓	0.144	**0.0164**	**0.63**	**0.039**	**13.28**
LayoutDM†		0.150	0.0192	0.41	0.190	27.09
LayoutDM†	✓	**0.123**	**0.0144**	**0.51**	**0.091**	**10.03**

Table 3. Retrieval augmentation for CGL-GAN and LayoutDM† on the PKU test split.

that can handle many constrained generation tasks. Since the model is originally designed for content-agnostic layout generation, we extend the model to accept an input image.
Autoreg Baseline is the one described in Sec. 3.2 and is equivalent to our RALF without retrieval augmentation.
RALF is our model described in Sec. 3.
Real Data is the ground truth, which can be considered the upper bound. Since we draw the sample from the test split, we calculate the FID score using the validation split.
Top-1 Retrieval is a nearest-neighbor layout without any generator, which can be considered a retrieval-only baseline.

4.4. Implementation Details

We re-implement most of the baselines since there are few official implementations publicly available, except for DS-GAN [18]. In RALF, we retrieve $K = 16$ nearest neighbor layouts. Following CGL-GAN [48], the height and width size of the input image are set to 350 and 240, respectively. We generate layouts on three independent trials and report the average of the metrics. We describe the details of training and network configuration in the supplementary material.

4.5. Unconstrained Generation

Baseline comparison. Table 1 presents the quantitative results on the annotated test split without user constraints. RALF achieves the best scores, except for the Occ metric of ICVT on CGL. Top-1 Retrieval, which almost disregards the given content, is unsuitable for the task, as we show deficient performance in content metrics.

Table 2 summarizes results on the unannotated test split. RALF achieves the best scores in all the metrics. Compared with Table 1, all the models exhibit slight performance degradation in PKU due to the domain gap problem [43] between inpainted canvases and clean canvases. We conjecture that the significant performance degradation in CGL comes from non-negligible spatial shifts in subject distributions, which we demonstrate in the supplementary material.

Effectiveness of retrieval augmentation. Tables 1 and 2 demonstrate that retrieval augmentation significantly enhances the Autoreg Baseline. The only exception is the Occ metric on CGL in Table 1, where the Autoreg Baseline already closely matches Real Data metrics.

Qualitative results. We show the qualitative comparison in Fig. 3. The results demonstrate that our RALF's ability to generate well-fitted, non-overlapping, and rational layouts. In contrast, the baseline methods often produce misaligned underlay embellishments and overlapped text elements as we indicate by red arrows. We also indicate undesirable elements that appear on a salient region by green arrows.

Training dataset size. Here, we show that retrieval augmentation is effective regardless of the training dataset size in Fig. 4. Notably, our RALF trained on just 3,000 samples outperforms the Autoreg Baseline trained on the full 7,734

Figure 3. Visual comparison of unconstrained generation with baselines. Input canvases are selected from the unannotated split.

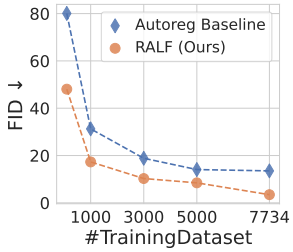

Figure 4. FID over the training dataset size (#TrainingDataset), which has up to 7,734 samples.

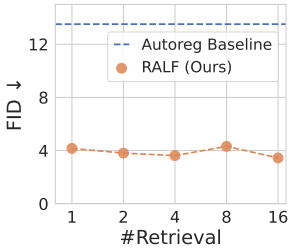

Figure 5. FID over the retrieval size K (#Retrieval).

Figure 6. Visual comparison of retrieval and generated layouts with different retrieval sizes ($K=1$ and 16). We display the top-4 examples for $K=16$ due to the limited space. The output layouts are generated using different random seeds for variety.

Train	Test	Method	Occ ↓	Rea ↓	Und ↑	Ove ↓
CGL	PKU	Autoreg Baseline	0.176	0.0276	0.84	0.037
		RALF (Ours)	**0.144**	**0.0249**	**0.96**	**0.023**
PKU	CGL	Autoreg Baseline	0.341	0.0464	0.29	0.037
		RALF (Ours)	**0.286**	**0.0355**	**0.79**	**0.036**

Table 4. Generation across the unannotated test splits. We train a model on PKU and then test it on CGL with the layout database of PKU, or vice versa.

samples in PKU.

Retrieval size K. We show that retrieval augmentation is not highly sensitive to the number of retrieved layouts K. As we plot in Fig. 5, retrieval augmentation significantly enhances the performance even with a *single* retrieved layout compared to the baseline. The plot indicates FID moderately gets better as we increase the retrieval size K.

We examine how different K affects the generated results in Fig. 6. The result of $K=1$ shows that the generated layout is similar to the reference layouts, while the result of $K=16$ shows that a variety of layouts are generated.

Retrieval augmentation for other generators. While our RALF is an autoregressive generator, we show that retrieval augmentation also benefits other generative models for content-aware layout generation. Here, we adapt CGL-GAN and LayoutDM† with retrieval augmentation and evaluate the performance. Table 3 summarizes the results. CGL-GAN and LayoutDM† combined with our retrieval augmentation consistently improve many evaluation metrics. We provide additional results in the supplementary material.

Out-of-domain generalization. Table 4 summarizes the

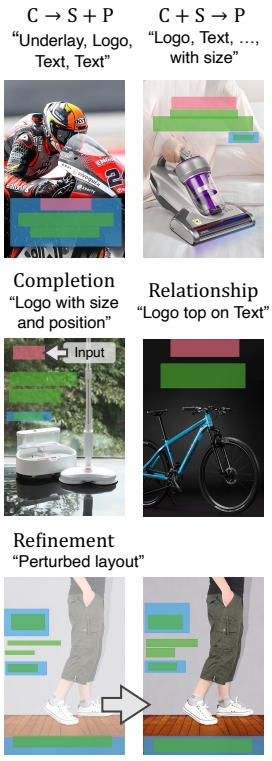

Figure 7. Examples of input constraints and generated results for each constrained generation task. Quotation marks indicate the constraints.

Method	PKU					CGL				
	Content		Graphic			Content		Graphic		
	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓
C → S + P										
CGL-GAN	0.132	0.0158	0.48	0.038	11.47	0.140	0.0213	0.65	0.047	23.93
LayoutDM†	0.152	0.0201	0.46	0.172	20.56	0.127	0.0192	0.79	0.026	3.39
Autoreg Baseline	0.135	0.0167	0.43	0.028	10.48	**0.124**	0.0188	0.89	0.015	1.36
RALF (Ours)	**0.124**	**0.0138**	**0.90**	**0.010**	**2.21**	0.126	**0.0180**	**0.97**	**0.006**	**0.50**
C + S → P										
CGL-GAN	0.129	0.0155	0.48	0.043	9.11	0.129	0.0202	0.75	0.027	6.96
LayoutDM†	0.143	0.0185	0.45	0.122	24.90	**0.127**	0.0190	0.82	0.021	2.18
Autoreg Baseline	0.137	0.0169	0.46	0.028	5.46	**0.127**	0.0191	0.88	0.013	0.47
RALF (Ours)	**0.125**	**0.0138**	**0.87**	**0.010**	**0.62**	0.128	**0.0185**	**0.96**	**0.006**	**0.21**
Completion										
CGL-GAN	0.150	0.0174	0.43	0.061	25.67	0.174	0.0231	0.21	0.182	78.44
LayoutDM†	0.135	0.0175	0.35	0.134	21.70	0.127	0.0192	0.76	0.020	3.19
Autoreg Baseline	0.125	0.0161	0.42	0.023	5.96	**0.124**	**0.0185**	0.91	0.011	2.33
RALF (Ours)	**0.120**	**0.0140**	**0.88**	**0.012**	**1.58**	0.126	**0.0185**	**0.96**	**0.005**	**1.04**
Refinement										
CGL-GAN	0.122	0.0141	0.39	0.090	6.40	**0.124**	0.0182	0.86	0.024	1.20
LayoutDM†	0.115	0.0121	0.57	0.008	2.86	0.127	0.0188	0.75	0.018	1.98
Autoreg Baseline	0.131	0.0171	0.41	0.026	5.89	0.126	0.0183	0.89	0.004	0.15
RALF (Ours)	**0.113**	**0.0109**	**0.95**	**0.004**	**0.13**	0.126	**0.0176**	**0.98**	**0.002**	**0.14**
Relationship										
Autoreg Baseline	0.140	0.0177	0.44	0.028	10.61	0.127	0.0189	0.88	0.015	1.28
RALF (Ours)	**0.122**	**0.0141**	**0.85**	**0.009**	**2.23**	**0.126**	**0.0184**	**0.95**	**0.006**	**0.55**

Table 5. Quantitative result of six constrained generation tasks on the PKU and CGL test split.

results of a cross-evaluation setup where we use different datasets for training and testing. For example, we use the database and training data from CGL and evaluate PKU in the upper half of Table 4. Remarkably, even in this out-of-domain setting, retrieval augmentation shows notable improvement and robust generalizability.

4.6. Constrained Generation

Following the task setup of content-agnostic generation [22], we evaluate several methods in the following constrained tasks in content-aware generation:
Category → Size + Position (C → S + P) takes in element types and generates the sizes and positions for each element.
Category + Size → Position (C + S → P) generates element positions based on given element categories and sizes.
Completion generates a complete layout using partially placed elements.
Refinement corrects cluttered layouts where elements are perturbed from the ground truth based on a normal distribution with mean 0 and standard deviation 0.01, following [39].
Relationship is conditioned on both element types and their spatial relationships, determined by the size and position of element pairs. We randomly use 10% of these relationships in our experiments, following [24].

Input constraints and generated examples for these tasks are illustrated in Fig. 7.

Baseline comparison. Table 5 summarizes constrained generation results. The results indicate that RALF is effective even for constrained generation tasks. For tasks such as C + S → P and Refinement, RALF shows notable improvement in the FID metric. This suggests that referencing authentic examples to understand element relationships enhances position prediction accuracy. Overall, the results highlight RALF's capability to significantly augment the generative performance over the baseline approach.

4.7. Ablation Study

We investigate our design choices in our retrieval augmentation proposed in Sec. 3.3.

Layout retrieval. We employ an image feature extractor to compute the similarity between canvases. We provide a brief overview of possible choices. *DreamSim* [12] captures diverse aspects of the similarity simultaneously. *LPIPS* [46] focuses on low-level appearance similarity. *CLIP* [38] focuses on semantic similarity. *Saliency* focuses on spatial similarity using the saliency map. We obtain embeddings

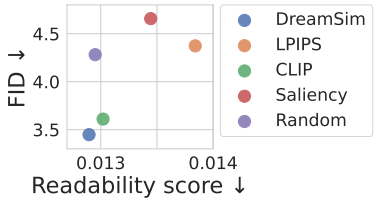

Figure 8. Comparison across different retrieval methods on the PKU test split. We report FID as the representative graphic metric and Readability score as the content metric.

for similarity computation by down-sampling and flattening S. *Random* serves as a naïve baseline by randomly sampling layouts without focusing on image similarity.

We train our RALF with each choice and assess the performance. Figure 8 plots FID and Readability score for each retrieval method, and Fig. 9 presents some retrieved examples. DreamSim shows the best balance in the graphic and content metrics. Random achieves a reasonable balance, suggesting that referring to real layouts is crucial. In comparison, we conjecture that increasing the chances of retrieving a more suitable reference further boosts the generation quality.

Feature augmentation. We explore the design of our feature augmentation module, as detailed in Table 6.
What types of features to fuse? RALF combines three features in Eq. (2). We observe that dropping some of the features, as in scenarios (B) and (C), leads to a slight deterioration of the performance. We try adding features of the top-K retrieved images $\tilde{f}_{\mathrm{I}} \in \mathbb{R}^{KH'W' \times d}$ that are encoded by the image encoder from the retrieved canvas. However, adding $\tilde{f}_{\mathrm{I}}$ results in decreased performance, as shown in (D).
Where to apply? Our model first applies the Transformer encoder and then retrieval augmentation to the image feature (A). We try another design (E), which places the augmentation module before the Transformer encoder, however, this results in worse readability and underlay metrics in exchange for the slight improvement in FID.

5. Discussion

Limitations. We acknowledge two limitations as follows: 1) Evaluation of content metrics: The current content metrics assume that well-designed layouts avoid placing elements over salient or cluttered areas. If a counterexample exists, the content metrics may not adequately measure layout quality. Also, the graphic metrics can be easily fooled by a real example, as evidenced by the FID score of the Top-1 baseline in Table 1. 2) Feature extraction of retrieved layouts: The layout encoder depends on the number of element categories in the dataset. For real-world creative scenarios, extending to an unlimited number of categories, *i.e.* an open-vocabulary setting [11], would be necessary.

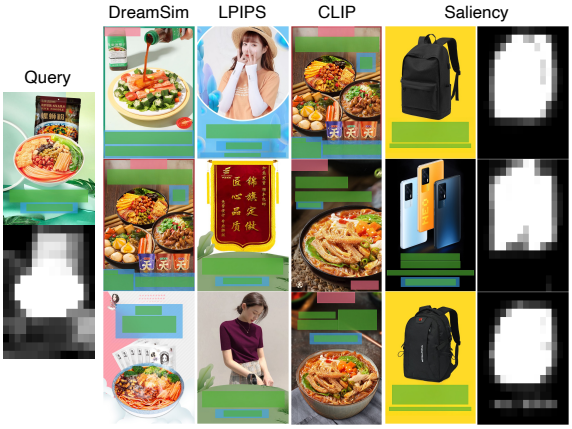

Figure 9. Qualitative comparison of different retrieval methods. We show the query and the top-3 retrieved examples for each method.

	Setting	Occ ↓	Rea ↓	Und ↑	Ove ↓	FID↓
A	Ours (Concatenate($f_{\mathrm{I}}, \tilde{f}_{\mathrm{L}}, f_{\mathrm{C}}$))	**0.119**	**0.0129**	0.92	0.008	3.45
	What types of features to fuse?					
B	Concatenate(f_{C},)	0.134	0.0144	0.92	0.008	4.67
C	Concatenate($f_{\mathrm{I}}, \tilde{f}_{\mathrm{L}}$)	0.123	0.0133	0.91	**0.007**	4.08
D	Concatenate($f_{\mathrm{I}}, \tilde{f}_{\mathrm{L}}, f_{\mathrm{C}}, \tilde{f}_{\mathrm{I}}$)	0.141	0.0148	**0.93**	0.009	8.82
	Where to apply?					
E	Before Trans enc	0.120	0.0138	0.72	0.009	**2.34**

Table 6. Ablation study of RALF design on the PKU test split. The top two results are highlighted in **bold** and underline, respectively. Features include the input canvas feature (f_{I}), retrieved layouts feature ($\tilde{f}_{\mathrm{L}}$), cross-attended feature (f_{C}), and retrieved images feature ($\tilde{f}_{\mathrm{I}}$). The full setting of our model (A) is described in Eq. (2).

Future work. We outline two prospective directions to enhance retrieval augmentation for content-aware generation further: 1) Ensemble approaches: integrating multiple retrieval results could potentially improve the generation quality. 2) Diversifying retrieval modalities: exploring layout retrieval using alternative modalities, such as language, could widen the application scope. Yet, generating a whole poster beyond bounding boxes, such as image content, text copies, or styling attributes, remains challenging due to the limited training data for layered graphic designs. Even for such a task, we expect that the retrieval augmentation approach could alleviate the data scarcity problem.

Potential societal impacts. As common in any generative models, our RALF may unintentionally produce counterfeit advertisements or magazine layouts, posing risks of deception and dissemination of misleading information.

Acknowledgement

We would like to thank Mayu Otani, Xueting Wang, Seiji Kurokoshi, and Atsushi Honda for their insightful feedback. This research was partly supported by JSPS KAKENHI 22KJ1014.

References

[1] Adobe Illustrator CC. https://www.adobe.com/products/illustrator.html, Last accessed 17 November, 2023. 1

[2] Maneesh Agrawala, Wilmot Li, and Floraine Berthouzoz. Design Principles for Visual Communication. *Communications of the ACM*, 54(4), 2011. 2

[3] Diego Martin Arroyo, Janis Postels, and Federico Tombari. Variational Transformer Networks for Layout Generation. In *CVPR*, 2021. 2

[4] Akari Asai, Sewon Min, Zexuan Zhong, and Danqi Chen. ACL 2023 Tutorial: Retrieval-based Language Models and Applications . *ACL*, 2023. 2

[5] Andreas Blattmann, Robin Rombach, Kaan Oktay, and Björn Ommer. Retrieval-Augmented Diffusion Models. In *NeurIPS*, 2022. 1, 2

[6] Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George van den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego de Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack W. Rae, Erich Elsen, and Laurent Sifre. Improving language models by retrieving from trillions of tokens. *arXiv preprint arXiv:2112.04426*, 2021. 1, 2

[7] Yunning Cao, Ye Ma, Min Zhou, Chuanbin Liu, Hongtao Xie, Tiezheng Ge, and Yuning Jiang. Geometry Aligned Variational Transformer for Image-conditioned Layout Generation. In *ACM MM*, 2022. 2, 3, 4, 5

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *ECCV*, 2020. 2

[9] Shang Chai, Liansheng Zhuang, and Fengying Yan. LayoutDM: Transformer-based Diffusion Model for Layout Generation. In *CVPR*, 2023. 2

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*, 2019. 2

[11] Weixi Feng, Wanrong Zhu, Tsu-jui Fu, Varun Jampani, Arjun Akula, Xuehai He, Sugato Basu, Xin Eric Wang, and William Yang Wang. LayoutGPT: Compositional Visual Planning and Generation with Large Language Models. In *NeurIPS*, 2023. 8

[12] Stephanie Fu*, Netanel Tamir*, Shobhita Sundaram*, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning New Dimensions of Human Visual Similarity using Synthetic Data. In *NeurIPS*, 2023. 4, 7

[13] Shunan Guo, Zhuochen Jin, Fuling Sun, Jingwen Li, Zhaorui Li, Yang Shi, and Nan Cao. Vinci: An Intelligent Graphic Design System for Generating Advertising Posters. In *CHI*, 2021. 1

[14] Kamal Gupta, Alessandro Achille, Justin Lazarow, Larry Davis, Vijay Mahadevan, and Abhinav Shrivastava. LayoutTransformer: Layout Generation and Completion with Self-attention. In *ICCV*, 2021. 2, 3

[15] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. REALM: Retrieval-Augmented Language Model Pre-Training. In *ICML*, 2020. 1, 2

[16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 3

[17] Scarlett R. Herring, Chia-Chen Chang, Jesse Krantzler, and Brian P. Bailey. Getting Inspired! Understanding How and Why Examples Are Used in Creative Design Practice. In *CHI*, 2009. 1

[18] Hsiao Yuan Hsu, Xiangteng He, Yuxin Peng, Hao Kong, and Qing Zhang. PosterLayout: A New Benchmark and Approach for Content-Aware Visual-Textual Presentation Layout. In *CVPR*, 2023. 1, 2, 3, 4, 5

[19] Naoto Inoue, Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. LayoutDM: Discrete Diffusion Model for Controllable Layout Generation. In *CVPR*, 2023. 1, 2, 4, 5

[20] Ali Jahanian, Jerry Liu, Qian Lin, Daniel Tretter, Eamonn O'Brien-Strain, Seungyon Claire Lee, Nic Lyons, and Jan Allebach. Recommendation System for Automatic Design of Magazine Covers. In *IUI*, 2013. 1

[21] Zhaoyun Jiang, Shizhao Sun, Jihua Zhu, Jian-Guang Lou, and Dongmei Zhang. Coarse-to-fine generative modeling for graphic layouts. In *AAAI*, 2022. 2

[22] Z. Jiang, J. Guo, S. Sun, H. Deng, Z. Wu, V. Mijovic, Z. Yang, J. Lou, and D. Zhang. LayoutFormer++: Conditional Graphic Layout Generation via Constraint Serialization and Decoding Space Restriction. In *CVPR*, 2023. 2, 3, 7

[23] Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. LayoutVAE: Stochastic Scene Layout Generation from a Label Set. In *CVPR*, 2019. 2

[24] Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. Constrained Graphic Layout Generation via Latent Optimization. In *ACM MM*, 2021. 2, 4, 7

[25] Xiang Kong, Lu Jiang, Huiwen Chang, Han Zhang, Yuan Hao, Haifeng Gong, and Irfan Essa. BLT: Bidirectional Layout Transformer for Controllable Layout Generation. In *ECCV*, 2022. 2

[26] Hsin-Ying Lee, Weilong Yang, Lu Jiang, Madison Le, Irfan Essa, Haifeng Gong, and Ming-Hsuan Yang. Neural Design Network: Graphic Layout Generation with Constraints. In *ECCV*, 2019. 4

[27] Elad Levi, Eli Brosh, Mykola Mykhailych, and Meir Perez. DLT: Conditioned Layout Generation with Joint Discrete-Continuous Diffusion Layout Transformer. In *ICCV*, 2023. 2

[28] Fengheng Li, An Liu, Wei Feng, Honghe Zhu, Yaoyu Li, Zheng Zhang, Jingjing Lv, Xin Zhu, Junjie Shen, Zhangang Lin, and Jingping Shao. Relation-Aware Diffusion Model for Controllable Poster Layout Generation. In *CIKM*, 2023. 2

[29] Jianan Li, Jimei Yang, Aaron Hertzmann, Jianming Zhang, and Tingfa Xu. LayoutGAN: Generating Graphic Layouts with Wireframe Discriminators. In *ICLR*, 2019. 2

[30] Jianan Li, Jimei Yang, Jianming Zhang, Chang Liu, Christina Wang, and Tingfa Xu. Attribute-Conditioned Layout GAN for Automatic Graphic Design. *IEEE TVCG*, 27(10), 2021. 2

[31] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature Pyramid Networks for Object Detection. In *CVPR*, 2017. 3

[32] Simon Lok and Steven Feiner. A Survey of Automated Layout Techniques for Information Presentations. In *SmartGraphics*, 2001. 2

[33] Peter O'Donovan, Aseem Agarwala, and Aaron Hertzmann. DesignScape: Design with Interactive Layout Suggestions. In *CHI*, 2015. 1, 2

[34] Chunyao Qian, Shizhao Sun, Weiwei Cui, Jian-Guang Lou, Haidong Zhang, and Dongmei Zhang. Retrieve-Then-Adapt: Example-based Automatic Generation for Proportion-related Infographics. *IEEE TVCG*, 27(2), 2021. 2

[35] Xuebin Qin, Zichen Zhang, Chenyang Huang, Chao Gao, Masood Dehghan, and Martin Jagersand. BASNet: Boundary-Aware Salient Object Detection. In *CVPR*, 2019. 2

[36] Xuebin Qin, Hang Dai, Xiaobin Hu, Deng-Ping Fan, Ling Shao, and Luc Van Gool. Highly Accurate Dichotomous Image Segmentation. In *ECCV*, 2022. 2

[37] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019. 3

[38] 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. In *ICML*, 2021. 4, 7

[39] Soliha Rahman, Vinoth Pandian Sermuga Pandian, and Matthias Jarke. RUITE: Refining UI Layout Aesthetics Using Transformer Encoder. In *IUI Companion*, 2021. 7

[40] Shelly Sheynin, Oron Ashual, Adam Polyak, Uriel Singer, Oran Gafni, Eliya Nachmani, and Yaniv Taigman. KNN-Diffusion: Image Generation via Large-Scale Retrieval. In *ICLR*, 2023. 1, 2

[41] Roman Suvorov, Elizaveta Logacheva, Anton Mashikhin, Anastasia Remizova, Arsenii Ashukha, Aleksei Silvestrov, Naejin Kong, Harshith Goka, Kiwoong Park, and Victor Lempitsky. Resolution-robust Large Mask Inpainting with Fourier Convolutions. In *WACV*, 2022. 4

[42] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017. 2

[43] Chenchen Xu, Min Zhou, Tiezheng Ge, Yuning Jiang, and Weiwei Xu. Unsupervised Domain Adaption With Pixel-Level Discriminator for Image-Aware Layout Generation. In *CVPR*, 2023. 5

[44] Xuyong Yang, Tao Mei, Ying-Qing Xu, Yong Rui, and Shipeng Li. Automatic Generation of Visual-Textual Presentation Layout. *ACM TOMM*, 12(2), 2016. 1, 2

[45] Junyi Zhang, Jiaqi Guo, Shizhao Sun, Jian-Guang Lou, and Dongmei Zhang. LayoutDiffusion: Improving Graphic Layout Generation by Discrete Diffusion Probabilistic Models. In *ICCV*, 2023. 2

[46] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 7

[47] Xinru Zheng, Xiaotian Qiao, Ying Cao, and Rynson W. H. Lau. Content-Aware Generative Modeling of Graphic Design Layouts. *ACM TOG*, 38(4), 2019. 1, 2

[48] Min Zhou, Chenchen Xu, Ye Ma, Tiezheng Ge, Yuning Jiang, and Weiwei Xu. Composition-aware Graphic Layout GAN for Visual-textual Presentation Designs. In *IJCAI*, 2022. 1, 2, 3, 4, 5