Frontiers of Information Technology & Electronic Engineering
www.jzus.zju.edu.cn; engineering.cae.cn; www.springerlink.com
ISSN 2095-9184 (print); ISSN 2095-9230 (online)
E-mail: jzus@zju.edu.cn

Iris: a multi-constraint graphic layout generation system*#

Liuqing CHEN‡[1,2], Qianzhi JING[1], Yixin TSANG[1], Tingting ZHOU[3]

[1]*College of Computer Science and Technology, Zhejiang University, Hangzhou 310030, China*

[2]*Zhejiang–Singapore Innovation and AI Joint Research Lab, Hangzhou 310058, China*

[3]*Alibaba Group, Hangzhou 310034, China*

E-mail: chenlq@zju.edu.cn; jingqz@zju.edu.cn; tsangeyan@zju.edu.cn; miaojing@taobao.com

Received Apr. 30, 2023; Revision accepted Nov. 9, 2023; Crosschecked Apr. 2, 2024

Abstract: In graphic design, layout is a result of the interaction between the design elements in the foreground and background images. However, prevalent research focuses on enhancing the quality of layout generation algorithms, overlooking the interaction and controllability that are essential for designers when applying these methods in real-world situations. This paper proposes a user-centered layout design system, Iris, which provides designers with an interactive environment to expedite the workflow, and this environment encompasses the features of user-constraint specification, layout generation, custom editing, and final rendering. To satisfy the multiple constraints specified by designers, we introduce a novel generation model, multi-constraint LayoutVQ-VAE, for advancing layout generation under intra- and inter-domain constraints. Qualitative and quantitative experiments on our proposed model indicate that it outperforms or is comparable to prevalent state-of-the-art models in multiple aspects. User studies on Iris further demonstrate that the system significantly enhances design efficiency while achieving human-like layout designs.

Key words: Graphic layout generation; Deep generative model; Layout design system
https://doi.org/10.1631/FITEE.2300312 **CLC number:** TP302

1 Introduction

Graphic design skillfully integrates diverse design components (e.g., images, text, and decorations) to create visually engaging presentations that instantly draw attention and convey information. It is widely applied in media of various types, such as product posters, magazines, and interactive application interfaces. Layout design serves as the core of graphic design, aiming to arrange all design elements logically, including the determination of each element's spatial position and dimension. The placement and size of design elements not only depend on their relationship with other elements in the foreground but are also significantly influenced by background images, particularly in poster and billboard design. The creation of high-quality layout design demands considerable input from designers in terms of their expertise, aesthetic judgment, and time, thus indicating the need for an automated tool capable of achieving efficient, top-notch layout creation.

Recent studies have investigated the use of deep generative models like generative adversarial networks (GANs) (Li et al., 2019; Kikuchi et al., 2021; Zhou et al., 2022), variational autoencoders (VAEs)

‡ Corresponding author

* Project supported by the Alibaba–Zhejiang University Joint Research Institute of Frontier Technologies, China and the Zhejiang–Singapore Innovation and AI Joint Research Lab, China

Electronic supplementary materials: The online version of this article (https://doi.org/10.1631/FITEE.2300312) contains supplementary materials, which are available to authorized users

ORCID: Liuqing CHEN, https://orcid.org/0000-0002-9049-0394

(Arroyo et al., 2021; Cao et al., 2022), and diffusion models (Hui et al., 2023; Inoue et al., 2023) for addressing layout generation challenges. Some approaches have also demonstrated that the self-attention mechanism in Transformers can aid in modeling relationships between elements (Arroyo et al., 2021; Gupta et al., 2021). These methods typically depict a layout as a sequence comprising all design elements within the layout, generating the layout by predicting each element's position and size attributes.

Nonetheless, these studies are centered primarily on improving the quality of the generated layouts, overlooking the requirements of designers in real-world situations. In practical work, the first step is for the designers to identify the specific design elements to be displayed. Following this, they proceed to create an initial draft of the layout based on these identified elements. This draft then undergoes multiple rounds of iteration and refinement. Ultimately, through rendering, the final graphic design is realized. Despite the ability of prevalent methods to swiftly generate a multitude of layouts, they fall short in providing an interactive platform that aids designers in efficiently completing this design process. To address this challenge, we introduce a user-centered layout design system, Iris, which seamlessly integrates layout generation models within an interactive design environment, facilitating a comprehensive workflow that encompasses user-constraint specification, layout generation, custom editing, and final rendering. It is accessible to a broad range of users including designers and non-professionals, and hosts a wide array of design scenarios, including user interfaces, posters, magazines, and product display cards.

Moreover, concerning the layout generation model employed by this system, while current methods are capable of generating layouts with high quality, they fail to satisfy the precise constraints stipulated by designers. Designers typically need to create layouts with intra- and inter-domain constraints. For instance, in a poster design task shown in Fig. 1, the product subject to be showcased is fixed, serving as the inter-domain constraint in the form of a background image. Concurrently, the quantity and category of design elements employed to convey supplementary subject information (e.g., text and headings) or augment visual appeal (e.g., embellishments and underlays) are pre-determined, generally forming the intra-domain constraint represented as a sequence of design element categories. Previous studies are limited to generating layouts either randomly (Li et al., 2019; Arroyo et al., 2021; Gupta et al., 2021) or with constraints from a single domain (Kikuchi et al., 2021; Zhou et al., 2022; Inoue et al., 2023; Xu et al., 2023). Here, we propose a novel layout generation model, denoted as multi-constraint LayoutVQ-VAE, to tackle the efficient generation of layouts that satisfy multiple constraints, supporting the layout generation functionality of Iris.

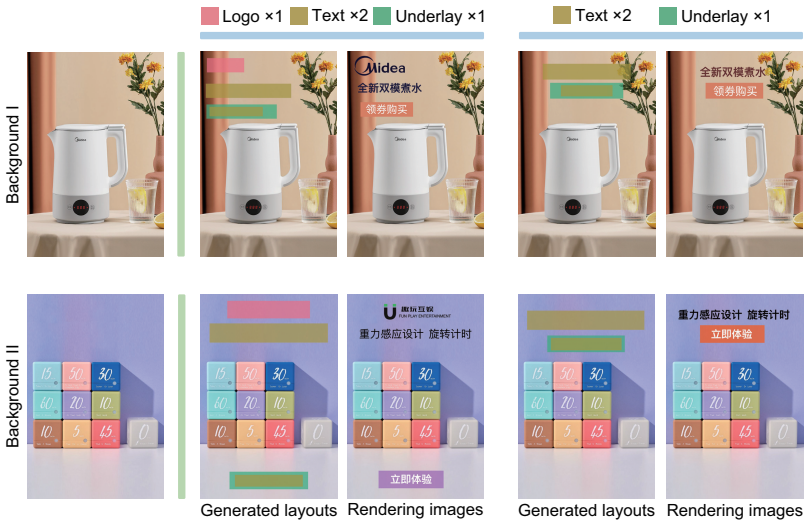

Fig. 1 Examples of layout generation with multiple constraints. Given a background image and the categories of design elements to be presented, our model generates various layouts (left) that fit the constraints, and our intelligent system synthesizes rendering images (right) by combining layouts and user-input design elements

Layout generation with multiple constraints involves the arrangement of user-specified elements on a background image in a sensible manner, without obscuring the primary subject, while simultaneously satisfying standard aesthetic expectations. This is more challenging than unconditional or single-condition-constrained layout generation, as these constraints originate from various domains and have different modalities. To represent constraints from multiple domains in a general manner while retaining crucial features like element quantities, categories, subject's spatial information, and smooth regions within background images, we propose a serialized constraint definition format. Specifically, the intra-domain constraint (the sequence of user-specified element categories) is discrete, and we also represent the inter-domain constraint (a user-specified background image) as a image patch sequence, which provides more positional information than pixel-level representation. Previous methods primarily employed continuous features for layout representation (Kikuchi et al., 2021; Cao et al., 2022; Zhou et al., 2022), creating a data representation discrepancy between continuous features and discrete constraints, which hinders models from effectively learning their relationships. Thus, we innovatively discretize the geometric parameters of layout elements and use the VQ-VAE framework (van den Oord et al., 2017) to learn the discrete latent representation of layouts. As a result, the powerful Transformer can be used to model the relationship between layout distributions and multiple constraints.

The contributions made by the present research can be summarized as follows:

1. We propose the multi-constraint LayoutVQ-VAE for layout generation constrained on element categories and the background image. The model learns a discrete latent representation of layouts and uniformly serializes constraints from different domains, which is a novel approach that facilitates modeling the relationship between layouts and multiple constraints.

2. We propose Iris, a user-centered, intelligent layout generation system. It assists designers throughout the workflow, covering input specification, layout generation, custom editing, and final rendering, and accommodates a wide variety of design scenarios, including posters, magazines, and user interfaces.

3. Experiments conducted on multiple datasets show that our model outperforms or competes favorably with state-of-the-art models. User studies carried out with our intelligent system, Iris, validate its ability to rival human designers in diverse design tasks while substantially improving design efficiency.

2 Related works

2.1 Layout generation models

Earlier approaches to layout generation typically relied on templates, dynamic programming, or heuristic design rules (Jacobs et al., 2003; Schrier et al., 2008; O'Donovan et al., 2014). Over the past few years, a substantial quantity of research has employed deep generative models to tackle layout generation challenges. Table 1 presents a comprehensive comparative analysis of prevalent layout generation models from multiple perspectives. LayoutGAN (Li et al., 2019) pioneers the use of deep generative models in this task, employing design element sequences to represent layouts and a GAN to model the layout distribution. However, this model necessitates a fixed number of elements and yields suboptimal layout quality. To enhance the quality, LayoutTransformer (Gupta et al., 2021) showcases the benefits of self-attention in modeling relationships among layout elements and demonstrates that parameter discretization could improve aesthetic metrics. Despite these advancements, the generation process lacks theoretical guarantees for capturing the full diversity of layout distributions caused by the autoregressive model. To tackle this, VTN (Arroyo et al., 2021) employs a self-attention-based VAE to approximate the layout distribution. Jiang et al. (2022) proposed a coarse-to-fine approach and decomposed the generation process into two stages to improve the quality of the generated layouts with a lot of elements. Although these methods gradually improve the layout generation quality and diversity, their generation processes remain uncontrollable.

Recent advances in constrained layout generation can be categorized broadly into two groups: intra-domain and inter-domain constraint-based generation. Intra-domain constraints relate to the internal properties of elements within the layout. AC LayoutGAN (Li et al., 2021) extends the original LayoutGAN by constraining the element area,

Table 1 Comprehensive comparison of representative layout generation models

Layout generation model	Intra-domain constraint	Inter-domain constraint	Framework	Layout representation
LayoutGAN (Li et al., 2019)	✗	✗	GAN	Continuous latent variables
LayoutTransformer (Gupta et al., 2021)	✗	✗	Transformer	Discrete variables*
VTN (Arroyo et al., 2021)	✗	✗	Transformer-based VAE	Continuous latent variables
Jiang et al. (2022)'s	✗	✗	Transformer-based VAE	Continuous latent variables
AC LayoutGAN (Li et al., 2021)	✓	✗	GAN	Continuous latent variables
LayoutGAN++ (Kikuchi et al., 2021)	✓	✗	Transformer-based GAN	Continuous latent variables
BLT (Kong et al., 2022)	✓	✗	Transformer	Discrete variables*
LayoutFormer++ (Jiang et al., 2023)	✓	✗	Transformer	Discrete variables*
LDGM (Hui et al., 2023)	✓	✗	Transformer-based diffusion model	Discrete variables*
LayoutDM (Inoue et al., 2023)	✓	✗	Transformer-based diffusion model	Discrete variables*
ContentGAN (Zheng et al., 2019)	✗	✓	VAE-GAN	Continuous latent variables
CGL-GAN (Zhou et al., 2022)	✗	✓	Transformer-based GAN	Continuous variables**
ICVT (Cao et al., 2022)	✗	✓	Transformer-based VAE	Continuous latent variables
DS-GAN (Hsu et al., 2023)	✗	✓	LSTM-based GAN	Continuous variables**
PGL-GAN (Xu et al., 2023)	✗	✓	Transformer-based GAN	Continuous variables**
Multi-constraint LayoutVQ-VAE (ours)	✓	✓	Transformer-based VQ-VAE	Discrete latent variables

* Layouts using discrete variables at the element attribute level; ** layouts using continuous variables at the element attribute level. GAN: generative adversarial network; VAE: variational autoencoder; LSTM: long short-term memory; VQ-VAE: vector-quantized variational autoencoder

aspect ratio, and reading order. LayoutGAN++ (Kikuchi et al., 2021) employs a conditional GAN (CGAN) architecture conditioned on element types and quantities, and introduces the CLG-LO algorithm to optimize the generated layouts based on user-specified constraints. BLT (Kong et al., 2022) leverages a bidirectional Transformer for controllable generation of element geometric parameters. The latest studies focus on handling diverse intra-domain constraints flexibly and uniformly. LayoutFormer++ (Jiang et al., 2023) proposes a constraint serialization scheme and formulates conditional layout generation as a sequence-to-sequence transformation. LDGM (Hui et al., 2023) and LayoutDM (Inoue et al., 2023) leverage diffusion models but adopt distinct approaches to incorporate intra-domain constraints; LDGM uses a general task setting and treats incomplete layouts as intermediary states, while LayoutDM employs a logical adjustment strategy during inference without altering the pre-trained model. Although these two models employ discrete variables to represent layouts, their usage is primarily confined to capturing element attributes. In contrast to the approaches adopted in these models, our model extracts discrete latent variables from these element attributes to comprehensively represent the entire layout.

On the other hand, inter-domain constraints involve external factors that impact the overall layout arrangement, typically introduced by background images or visual and textual semantics of elements. ContentGAN (Zheng et al., 2019) initially explores text and image semantic-driven layout generation but produces pixel-level layouts that necessitate subsequent element identification and adjustment. CGL-GAN (Zhou et al., 2022) and ICVT (Cao et al., 2022) generate product posters using background images as constraints. Recent research strongly emphasizes the importance of accurate extraction of background image features. DS-GAN (Hsu et al., 2023) and PGL-GAN (Xu et al., 2023) use convolutional neural networks (CNNs) to extract features. In this context, DS-GAN employs ResNet-FPN to extract the image saliency map, while PGL-GAN uses multiple upsampling layers to capture the shallow level feature map of images. However, most of these methods primarily focus on handling layout generation under individual constraints. To our knowledge, there is no research that tackles the more realistic scenario where both types of constraints exist. Our model, in addressing this challenge, represents a forward step in the literature on the topic.

2.2 Automated layout design systems

As graphic layout design becomes more prevalent, numerous design systems have been developed to assist designers. However, most of these systems are unable to generate intelligent layouts autonomously, or they can produce only template-based layouts, resulting in limited diversity. Table 2 facilitates a clear identification of distinct features and functionalities across various design systems.

Table 2 Comprehensive comparison of some representative design systems

System	Automatic generation	Layout input	Element definability*	Layout generation method	Understanding background image semantics	Editability	Use case
Figma	✗	–	✓	–	–	✓	2D design
GRIDS	✓	Layout elements	✗	MILP	✗	✓	UI
Luban	✓	Images, text	✗	Template prediction	✗	✓	Posters
Vinci	✓	Images, text	✗	Template prediction	✗	✓	Posters
Midjourney	✓	Text prompt	✗	Generation algorithm	✓	✗	2D design
Iris (ours)	✓	Images, number and category of elements	✓	Transformer-based VQ-VAE	✓	✓	UI, posters, PDCard**, magazines, documents

* Indicating whether it can be designed according to the given number and category of elements; ** product card layout design for mobile shopping Apps. MILP: mixed-integer linear programming; VQ-VAE: vector-quantized variational autoencoder; UI: user interface

Figma lacks automated generation capability. Vinci (Guo et al., 2021) and Luban (You et al., 2020) systems primarily rely on template prediction for poster generation, while they do not fully understand the semantics of the background image, restricting their adaptability. While GRIDS (Dayama et al., 2020) employs a rigorous algorithm like mixed-integer linear programming for layout generation, its usability is significantly hampered by complex element grouping and constraints in its applications during the design phase. In contrast, Iris demonstrates an efficient, automated, and customized approach. It allows designers to simultaneously define the layout background, as well as the number and category of design elements. By using generation models, it learns the relationship between these constraints and the layout, resulting in automatic layout generation.

There is research that has experimented with the latest large-scale text-to-image generation models, like Midjourney, specifically for designing posters and billboards. However, Midjourney focuses on generating semantic content for posters at the pixel level. The quality of its generated poster is reliant heavily on user-input prompts, and there is a lack of editable features. Iris, on the other hand, aims to offer designers a user-friendly and controllable system for creating poster layouts. To comprehensively compare Midjourney and our system, we conduct a poster layout design task using both systems. More information can be found in the supplementary materials.

3 Multi-constraint LayoutVQ-VAE

In this section, we introduce the proposed model of multi-constraint LayoutVQ-VAE. First, we briefly revisit the concept of VQ-VAE (van den Oord et al., 2017). Subsequently, we discuss the problem formulation, model framework, and how this framework combines VQ-VAE and Transformer (Vaswani et al., 2017) to model the distribution of layouts conditioned on the background image and element categories.

3.1 Vector-quantized variational autoencoder

The vector-quantized variational autoencoder (VQ-VAE) is a type of generation model that combines the ideas of a VAE and vector quantization (VQ). Its encoder learns to map input data (e.g., layouts) to a continuous latent space, while the decoder reconstructs the input data from the latent representation. The key difference between VQ-VAE and a traditional VAE is that, in the case of the former, the encoder output $\boldsymbol{z}_{\mathrm{e}}(x) \in \mathbb{R}^D$ is passed through a discretization bottleneck using a nearest neighbor lookup on embedding vectors $\boldsymbol{e} = [\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_K] \in \mathbb{R}^{K \times D}$, resulting in a discrete latent representation. Specifically, the decoder input $\boldsymbol{z}_{\mathrm{q}}(x)$ is defined as

$$\boldsymbol{z}_{\mathrm{q}}(x) = \boldsymbol{e}_k,\ k = \operatorname{argmin}_i \left\| \boldsymbol{z}_{\mathrm{e}} - \boldsymbol{e}_i \right\|_2, \qquad (1)$$

where $\|\cdot\|_2$ denotes the Euclidean norm and $i \in \{1, 2, \cdots, K\}$.

The corresponding quantization result can be represented by the index k of the embedding vector closest to $\boldsymbol{z}_{\mathrm{e}}(x)$. Then we obtain the discrete latent variable $z \in \{0, 1, \ldots, K-1\}$, and the posterior distribution $q(z|x)$ actually becomes a one-hot distribution. Through the prior definition of a simple uniform distribution over z, we obtain a Kullback–Leibler (KL) divergence constant that is equal to $\log K$, and once a good discrete latent structure of a modality is discovered with the use of VQ-VAE, it then becomes possible to train a powerful $p(z)$ over

these discrete random variables in a prior manner, thus yielding a diversity of interesting samples.

3.2 Problem formulation

Suppose that a layout dataset (X, Y) consists of N samples of layout x and its background image y. The graphic layout x consists of n design elements and can be defined as $x = \{\boldsymbol{c}_i, \boldsymbol{b}_i\}_{i=1}^{n}$, where $\boldsymbol{c}_i$ denotes the category of the i^{th} element (e.g., logo and text), and $\boldsymbol{b}_i = \left[b_i^x, b_i^y, b_i^w, b_i^h\right]$ represents the geometric parameter of its bounding box, including the center coordinates $[b_i^x, b_i^y]$ and the size $\left[b_i^w, b_i^h\right]$ (here, w and h represent the width and height, respectively). By following the approach adopted in Gupta et al. (2021), we apply an eight-bit uniform quantization on each geometric parameter to discretize their float values, which has been proven to facilitate the layout symmetry and alignment of elements. For instance, b^x after the uniform quantization becomes $\{b^x | b^x \in \mathbb{Z}, 0 \leq b^x \leq 255\}$. Then we concatenate all the bounding boxes into a flattened sequence as $\boldsymbol{b} = \left[b_1^x, b_1^y, b_1^w, b_1^h, b_2^x, b_2^y, b_2^w, b_2^h, \ldots, b_n^x, b_n^y, b_n^w, b_n^h\right]$, allowing us to use the attention mechanism for modeling the positional relationship among the bounding boxes. Element categories, which can be represented originally as discrete vectors, are also concatenated into a category sequence $\boldsymbol{c} = [\boldsymbol{c}_1, \boldsymbol{c}_2, \ldots, \boldsymbol{c}_n]$. Now, our problem can be formulated in terms of modeling the probability distribution of layouts X under the constraints image Y and element categories C.

We assume that layout x can be generated by a random process involving a discrete latent variable z: (1) z is generated from the conditional distribution $p_\theta(z|C, Y)$; (2) x is then generated from $p_\psi(x|z)$. Because $q_\phi(z|x)$ in VQ-VAE is the one-hot distribution on z, the log-likelihood of the complete model can be written as

$$\begin{aligned}\log p_{\theta,\psi}(X|C,Y) &= \log\left(p_\psi(X|z;\psi)p_\theta(z|C,Y)\right)\\ &\geq -\left(\mathop{\mathbb{E}}_{q_\phi(z|x)}\left[-\log p_\psi(x|z)\right] - \log p_\theta(z|C,Y)\right).\end{aligned} \tag{2}$$

Then we can divide the learning process into two stages: in stage 1, the encoder ϕ and decoder ψ learn the discrete latent representation of layout by minimizing the reconstruction loss, and in stage 2, Transformer θ models the conditional prior distribution by optimizing the negative log-likelihood (NLL) loss.

3.3 Model architecture

As shown in Fig. 2, our proposed model, multi-constraint LayoutVQ-VAE, contains two main components, namely a VQ-VAE to extract layout discrete latents and a Transformer to model the conditional distribution on the discrete latents.

3.3.1 Layout discrete latent variables

To learn the discrete latent variables of layouts X, we train a VQ-VAE where the attention layers in Transformers are applied as the backbone of the encoder and decoder.

Our encoder $q_\phi(\boldsymbol{z}_e|\boldsymbol{b}, \boldsymbol{c})$ takes the bounding boxes $\boldsymbol{b}$ and categories $\boldsymbol{c}$ of design elements in a

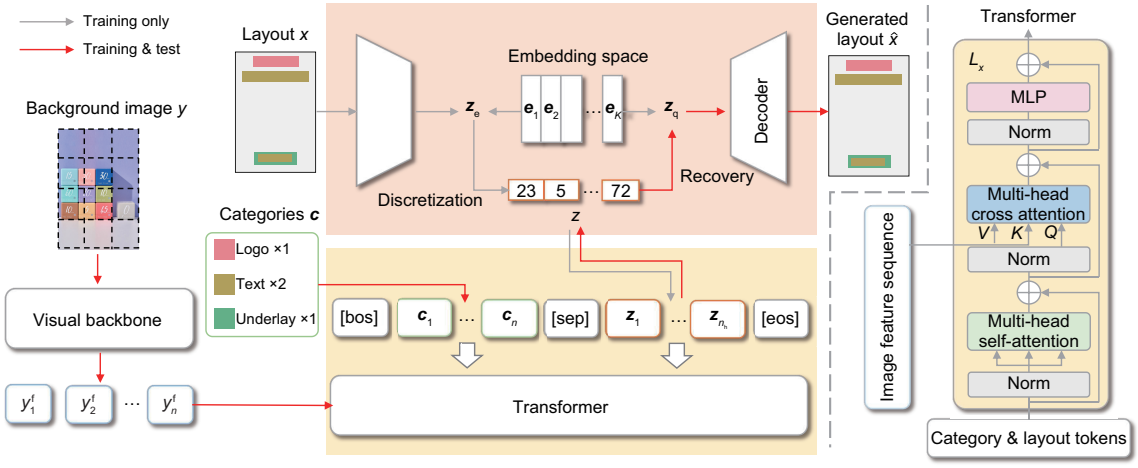

Fig. 2 Overview of multi-constraint LayoutVQ-VAE (MLP: multi-layer perceptron. References to color refer to the online version of this figure)

layout as the input and outputs the corresponding feature vector to represent this layout. Specifically, we first use a multi-layer perceptron (MLP) to project each input item to a D-dimensional space, and the resultant information elements are summed up with the position embedding, enabling us to thereby obtain the hidden input of the Transformer block. Next, taking inspiration from the CLS token in BERT (Devlin et al., 2019), we prepend several learnable embeddings to the hidden input sequence and pass the entire sequence through a Transformer encoder. The self-attention layers within it can uncover relationships between layout elements and incorporate information from the whole element sequence into the output vectors corresponding to the learnable embeddings. Ultimately, following the completion of the operation of the Transformer block, we retain only the output that is available in the form of learnable embeddings, and pass this output through an MLP to obtain the multi-head layout representation $\boldsymbol{z}_{\mathrm{e}}=\{\boldsymbol{z}_{\mathrm{e}}^{j}\}_{j=1}^{n_{\mathrm{h}}}$, where n_{h} represents the number of layout heads (i.e., learnable embeddings) and $\boldsymbol{z}_{\mathrm{e}}^{j}\in\mathbb{R}^{D}$.

By passing the encoder's output through a discrete bottleneck, each D-dimensional vector $\boldsymbol{z}_{\mathrm{e}}^{j}$ is quantized to a nearest embedding in a learnable shared codebook $\boldsymbol{e}=[\boldsymbol{e}_1,\boldsymbol{e}_2,\ldots,\boldsymbol{e}_K]\in\mathbb{R}^{K\times D}$ as in Eq. (1), where K is the size of the codebook. The quantization result can be represented by n_{h} indices of embeddings, and then we obtain the discrete latent variable $z\in\{0,1,\ldots,K-1\}^{n_{\mathrm{h}}}$ and the decoder input $\boldsymbol{z}_{\mathrm{q}}=\{\boldsymbol{z}_{\mathrm{q}}^{j}\}_{j=1}^{n_{\mathrm{h}}}$.

Our decoder $p_{\psi}(\boldsymbol{b},\boldsymbol{c}|z)$ has a similar architecture with the encoder. It takes $\boldsymbol{z}_{\mathrm{q}}$ and learnable position encoding as the input, processes them through an MLP, and subsequently feeds them into a bidirectional, non-autoregressive Transformer encoder. Ultimately, it reconstructs the categories $\boldsymbol{c}$ and bounding boxes $\boldsymbol{b}$ for each design element within the layout. The bidirectional, non-autoregressive structure is employed as it bidirectionally considers the influence of both preceding and succeeding elements on the current element and generates all elements simultaneously, rather than solely preceding elements like the autoregressive one.

3.3.2 Constraints and prior

To accurately model the relationship between layouts and constraints, it is crucial to adopt appropriate methods for constraint representation. Intra-domain constraints (i.e., user-specified element categories) are discrete and can be represented by token sequences, while inter-domain constraints (i.e., background images) lead to a more complex case. Previous studies typically used a CNN to extract image features (Zheng et al., 2019; Zhou et al., 2022). However, this method neglects the position information within the image, which is vital for predicting the positions of design elements in the layout. To tackle this challenge, we employ ViT (Dosovitskiy et al., 2021) as the visual backbone for extracting image features, since it operates on image patches rather than pixels. Image patches enable the model to more easily understand the relative positional information between them. Given an image $\boldsymbol{y}\in\mathbb{R}^{H\times W\times M}$, we divide it into several patches of a fixed size and concatenate these patches into a flattened two-dimensional (2D) sequence $\boldsymbol{y}_{\mathrm{p}}\in\mathbb{R}^{N\times\left(P^2\times M\right)}$, where $N=H\times W/P^2$ represents the number of patches, P denotes the size of the patches, M refers to the number of image channels, and H and W represent the height and width of the image respectively. Subsequently, the image patch sequence is fed into the ViT, and the hidden output of its Transformer block is obtained, serving as the image feature sequence $\boldsymbol{y}^{\mathrm{f}}\in\mathbb{R}^{N\times d_{\mathrm{I}}}$, where d_{I} represent the dimension of the image feature.

Since the prior distribution over the discrete layout latents $p(z)$ is a categorical distribution and both types of constraints are represented sequentially, we use an autoregressive Transformer decoder to model the conditional prior distribution as

$$p_{\theta}(z|C,Y)=\prod_{j=1}^{n_{\mathrm{h}}}p_{\theta}\left(z_j|z_{j-1},C,Y\right). \tag{3}$$

We concatenate the layout latent variable z and its corresponding element category sequence $\boldsymbol{c}$ into a single sequence input for the model, using self-attention layers to model their relationships. Simultaneously, we input the corresponding image feature sequence $\boldsymbol{y}^{\mathrm{f}}$ separately, employing cross-attention layers to model its relationship with the layout latents. Three special tokens <bos>, <eos>, and <sep> are added to the concatenated sequence to denote the beginning and end of the sequence, as well as the division between the element category sequence and the layout latent sequence. Learnable position embeddings are also added to the image patch sequence, enabling the model to effectively

capture spatial information within the image. The Transformer block consists of L attention layers, each comprising a multi-head self-attention layer, a multi-head cross-attention layer, and a fully connected feed-forward layer. Each sublayer adds residual connections (He et al., 2016) and LayerNorm (Ba et al., 2016).

3.4 Model learning and inference

In stage 1, since we learn the discrete latent representation using a VQ-VAE, the encoder ϕ and decoder ψ are optimized by minimizing

$$L_1 = \underbrace{\log p_\psi(x \mid \boldsymbol{z}_\mathrm{q}(x))}_{\text{Reconstruction loss}} + \underbrace{\beta \left\| \boldsymbol{z}_\mathrm{e}(x) - \mathrm{sg}\left(\boldsymbol{z}_\mathrm{q}(x)\right)\right\|_2^2}_{\text{Commitment loss}}, \tag{4}$$

where β is the weight coefficient set to 0.25 in all of our experiments, and $\mathrm{sg}(\cdot)$ represents the stop gradient operator that is defined as

$$\mathrm{sg}(\boldsymbol{x}) = \begin{cases} \boldsymbol{x}, & \text{forward pass,} \\ \boldsymbol{0}, & \text{backward pass.} \end{cases} \tag{5}$$

The reconstruction loss is employed to optimize the encoder and decoder, with gradients bypassing the embedding space and directly propagated to the encoder during the backward pass. The embedding space is optimized using exponentially moving averages (EMAs), as proposed by Kaiser et al. (2018). The commitment loss is used to ensure that the output of the encoder remains committed to a chosen embedding from the embedding space, which can aid the model in learning stable and meaningful discrete representations.

In stage 2, Transformer θ is optimized by minimizing the NLL loss for z:

$$L_2 = -\log p_\theta(z|C, Y). \tag{6}$$

After completing stages 1 and 2, we can use the trained model to generate layouts based on the specified multiple constraints, as indicated by the red arrows in Fig. 2. The background image is represented as image sequence Y through the visual backbone, and the number and category of elements are represented as sequence C. Transformer θ takes Y and C as conditional inputs, and infers and outputs the corresponding discrete layout representation z. Using the embedding space $\boldsymbol{e}$, $\boldsymbol{z}_\mathrm{q}$ corresponding to z is recovered and then inputted into the decoder ψ. The decoder ultimately decodes the category and the geometric parameters of each element in the layout. After rendering, the generated layout is obtained.

3.5 Implementation details

We implement our multi-constraint LayoutVQ-VAE with PyTorch (Paszke et al., 2019). For Transformer blocks in the encoder and decoder, we use eight layers with an input/output size of 512 and a feed-forward representation size of 256 and eight multi-attention heads. Four layout heads are used to represent the layout. In the embedding space, we use $K = 512$ and $D = 64$. For the visual backbone, we employ a pre-trained ViT-B-16 to extract image features, where each image is divided into patches of size 16×16. Consequently, the length of the image patch sequence is 196, and $d_\mathrm{I} = 768$. For Transformer blocks in stage 2, we stack six layers with an input/output size of 1024, and use four multi-attention heads for the self-attention layer and cross-attention layer. We train the model using the Adam optimizer on an NVIDIA GPU (GeForce RTX 3090 Founders Edition).

4 Experiments

In this section, we discuss the qualitative and quantitative performance of our proposed model across various layout generation tasks.

4.1 Experimental setup

4.1.1 Metrics

We use three kinds of quantitative metrics to assess the generated layouts from various perspectives:

1. Visual quality metrics. We assess the visual quality of the generated layouts by measuring the distance between them and the ground truth, both at the feature level and the bounding box level, using the Frechet inception distance (FID) (Heusel et al., 2017) and the maximum intersection over union (MaxIoU) (Kikuchi et al., 2021), respectively.

2. Compositional quality metrics. Following the approach adopted in Zhou et al. (2022), we compute the readability of text elements and the presentation of the product in generated poster layouts. Text readability is determined by calculating x- and y-direction gradients of the background image area occupied by text elements without an underlay, using

the Sobel operator. For evaluating the product presentation, we input the original background image and the image with the generated layout mask into VGG16, and compute the distance between their logit scores.

3. Aesthetic quality metrics. Two metrics are used to assess the aesthetic quality of the generated layouts. Alignment evaluates the extent to which elements maintain alignment with each other, by either their edges or centers. Overlap determines the degree of overlap between elements that should not semantically overlap and other elements within the layout, as detailed in Li et al. (2019).

4.1.2 Datasets

We evaluated our model on three publicly available datasets of layouts for posters, documents, and mobilephone user interfaces (UIs). CGL (Zhou et al., 2022) contains ≥60 000 advertising posters collected from e-commerce platforms. Each poster contains a background image and several design elements from five different types (i.e., logo, text, underlay, embellishment, and highlighted text). PubLayNet (Zhong et al., 2019) contains ≥330 000 scientific document layouts crawled from the Internet. It is annotated with five different elements (i.e., text, title, figure, list, and table). Rico (Deka et al., 2017) provides UI designs for mobile applications. It contains ≥91 000 layouts with 27 categories (toolbar, text, icon, ...).

For the CGL dataset, we used 95% of the official training set for training, the rest for validation, and the official test set (1000 pure images from all types of products) for test. As the test set consists solely of background images without any ground truth layouts, we enlisted the help of two designers to provide annotations for the design element category labels on each image. For PubLayNet, we excluded layouts with more than nine elements and used ≥160 000 layouts of the official training split for training, the remaining ≥8000 layouts for validation, and the official validation split (i.e., ≥4000 layouts) for test. For Rico, we preserved only the 13 most frequent elements in the dataset following the approach discussed in Kikuchi et al. (2021) and excluded layouts with more than nine elements. In the remaining ≥20 000 layouts, 85% were used for training, 5% for validation, and 10% for test.

4.2 Layout generation with intra-domain constraints

We evaluate our model on the PubLayNet and Rico datasets for layout generation with intra-domain constraints, and compare its performance with those of models reported in the literature, namely LayoutTransformer (Gupta et al., 2021), LayoutGAN++ (Kikuchi et al., 2021), and the most recent model, LayoutDM (Inoue et al., 2023). All models are implemented based on the released codes. Note that LayoutTransformer autoregressively generates random layouts by predicting the element category—bounding box sequences. In this experiment, to include element category constraints in it, we replace the predicted element categories with the given categories from the test samples during its inference process. For LayoutGAN++, we neglect its optimization algorithm to ensure a fair comparison with the performance of its generation model. LayoutDM injects intra-domain constraints in the form of masking or logit adjustment during inference.

Table 3 shows the quantitative comparison results. In terms of visual quality, our model demonstrates superior performance or comparability to previous approaches in both FID and MaxIoU metrics, indicating that the layouts generated by our

Table 3 Quantitative comparison in layout generation with intra-domain constraints

Method	PubLayNet				Rico			
	FID↓	MaxIoU↑	Alignment↓	Overlap↓	FID↓	MaxIoU↑	Alignment↓	Overlap↓
LayoutTransformer	20.61	0.33	**0.0013**	**0.11**	10.13	0.36	0.0057	**0.63**
LayoutGAN++	23.72	<u>0.34</u>	0.0019	<u>0.16</u>	13.58	0.36	0.0060	<u>0.66</u>
LayoutDM	**14.19**	**0.39**	0.0019	<u>0.16</u>	<u>7.34</u>	<u>0.39</u>	<u>0.0038</u>	0.67
Ours	<u>20.25</u>	**0.39**	<u>0.0014</u>	0.22	**6.26**	**0.47**	**0.0029**	0.85
Real data	–	–	0.0004	0.0022	–	–	0.0026	0.51

The values displayed in bold represent the optimal results, whereas those underlined denote the suboptimal results. An upward arrow indicates that higher values represent better performance, while a downward arrow indicates that lower values represent better performance

proposed model are closer to the real layouts in terms of layout features and bounding box levels. We infer that this is due to our innovative adoption of discrete latent variables to represent layout features. Compared to continuous latent variables adopted by LayoutGAN++, discrete latent variables conform more naturally to the layout data modality (i.e., using token sequences to represent layouts) (van den Oord et al., 2017), enabling a more accurate characterization of layouts and facilitating the learning of constraint relationships between layout and element category sequences. In terms of aesthetic quality, our performance in the alignment metric is superior to or very close to that of the best previous model, LayoutTransformer, but the performance in the overlap is relatively inferior. Note that both our model and LayoutTransformer discretize the geometric parameters of element bounding boxes, while LayoutGAN++ directly uses normalized continuous values to represent them. This suggests that discretizing geometric parameters can help generated layouts adhere more strictly to the alignment design principle.

Fig. 3 presents the qualitative comparison results. As can be seen, our method is capable of planning more reasonably the entire space based on the given element categories and quantities, whereas LayoutTransformer and LayoutGAN++ often have large areas of whitespace in their results; this is particularly evident in the case of LayoutTransformer. This is because our model uses a non-autoregressive decoder during the decoding phase, which allows for the extraction of relationships among all elements and the generation of reasonable predictions. The quality of the layout generated by LayoutDM on the PubLayNet dataset is comparable to that of our model, but its performance on the Rico dataset is slightly inferior to ours, consistent with the quantitative results. The qualitative results demonstrate that our generated layouts perform better in terms of alignment, while some LayoutGAN++ samples have misalignment issues, consistent with the quantitative evaluation results.

4.3 Layout generation with intra- and inter-domain constraints

The experiment on layout generation with both element category and background image constraints was conducted on the poster dataset CGL. The baselines include an image-agnostic method, LayoutTransformer, and multiple image-aware methods, namely ContentGAN (Zheng et al., 2019), CGL-GAN (Zhou et al., 2022), and the latest model PDA-GAN (Xu et al., 2023). ContentGAN was implemented based on its released code, with additional post-processing algorithms to denoise and extract elements from the generated pixel-level layouts. The quantitative results of CGL-GAN and PDA-GAN are sourced from the literature. The computation methods for the evaluation metrics employed in the literature are consistent with those outlined in the present study. Note that none of the

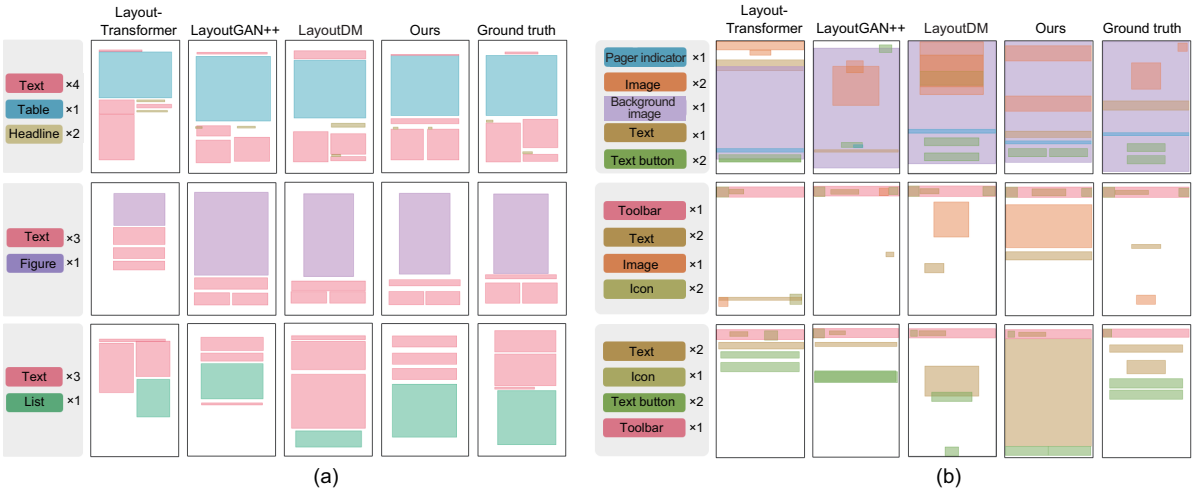

Fig. 3 Qualitative comparison in layout generation with intra-domain constraints: (a) PubLayNet; (b) Rico

three image-aware baselines considers intra-domain constraints since these methods lack the ability to control the quantity or category of design elements.

From the quantitative results presented in Table 4, it can be observed that our method outperforms the others in terms of text readability and product presentation, indicating that it can more accurately capture the relationship between the background image and layout. Our method performs better than other image-aware methods in terms of alignment metrics. However, for overlap, our method's performance is worse than those of LayoutTransformer and ContentGAN. Note that LayoutTransformer does not consider images, concentrating exclusively on the aesthetic aspects of the layout. We infer that generation models encounter a certain level of difficulty in achieving a balance between adhering to design rules and avoiding occlusion of the background subject. ContentGAN generates layout images at the pixel level. It removes noise from the image and identifies the positions of elements by extracting distinct color blocks. Therefore, its generated layouts have minimal overlapping occurrences.

The qualitative results shown in Fig. 4 demonstrate that our method can more accurately identify the product subject and smooth areas in the image. This capability enables us to arrange the given design elements in a judicious manner, achieving a compelling presentation effect while avoiding any obstruction to the products. This capability may be attributed to the fact that our model uses patch sequences to represent image features, thus enabling the preservation of the spatial information pertaining to different regions in the image, thereby providing a means for the guiding of element positions. Additionally, our method can accurately predict and arrange the given element categories, while other methods generate layouts with randomness and lack control over element categories and quantities. Overall, our approach skillfully strikes a balance between the impact of intra- and inter-domain constraints on layout generation.

4.4 Ablation study

We investigate whether an improved performance is obtained pursuant to the use of our visual backbone, ViT (as elucidated in Section 3.3.2), by substituting the image feature extraction method. After evaluating the prevalent methods, we opt for ResNet-FPN, an extensively used approach in previous works (Zhou et al., 2022; Hsu et al., 2023). This

Table 4 Quantitative comparison in layout generation with intra- and inter-domain constraints

Method	Text readability↓	Product presentation↓	Alignment↓	Overlap↓
LayoutTransformer	39.94	1.237	**0.0064**	0.0043
ContentGAN	36.88	1.010	0.0072	**0.0035**
CGL-GAN	34.01	0.816	0.0098	0.0256
PDA-GAN	33.55	0.688	0.0105	0.0290
Ours	**29.94**	**0.683**	**0.0064**	0.0206

The values displayed in bold represent the optimal results, whereas those underlined denote the suboptimal results. A downward arrow indicates that lower values represent better performance

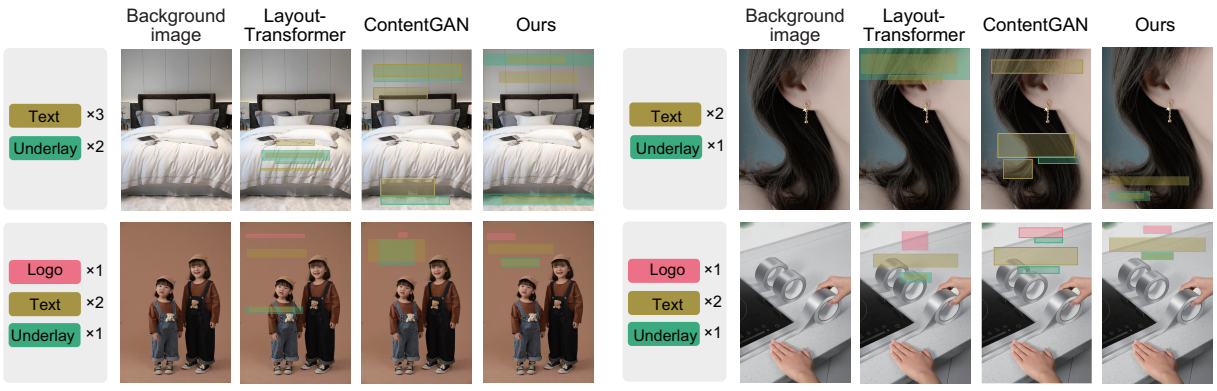

Fig. 4 Qualitative comparison in layout generation with intra- and inter-domain constraints

method first concatenates the original image and its saliency map, and then inputs the combined data into ResNet50 (He et al., 2016) to extract high-level features such as subject locations. It also leverages a multi-scale strategy on the last two convolutional blocks following FPN (Lin et al., 2017) to integrate lower-level features like region complexity by upsampling. The fused features are inputted into the generative models as the constraints.

The results are shown in Table 5. Compared to the model without a visual backbone, both ViT and ResNet-FPN significantly enhance text readability and the presentation of product subjects in the layout. Moreover, the employed ViT approach outperforms ResNet-FPN. This suggests that ViT, operating on image patches, better captures the positional information of product subject regions and smooth regions within the background image. In contrast, the ResNet-FPN approach extracts high-level semantics after multiple convolutional layers, potentially leading to the loss of certain positional information, subsequently affecting the quality of the generated layout.

5 Layout generation system

To provide greater assistance to designers in layout design tasks while simultaneously alleviating their workload, we propose Iris, an intelligent system that supports the automatic generation of layout in multiple design tasks. In this section, we describe the design goals of Iris and how Iris automates the generation of the layout design with the given constraints in various design tasks.

5.1 Design goals

In graphic design, the creation of effective layouts from elements like backgrounds, decorations, and text is a time-intensive task for designers, and is further compounded when producing high-quality layouts at large scale. To streamline this process, we introduce Iris, a Web-based tool leveraging our multi-constraint LayoutVQ-VAE for automated layout creation and real-time editing feedback. Iris is created to transcend mere algorithmic prowess, embodying a designer-centric system for a more streamlined, efficient, and user-friendly design workflow. From this perspective, the following goals are eventually established:

1. Amplified design efficiency. With its automated layout generation capability, Iris rapidly generates an array of layouts, offering designers diverse design options. This abundance not only sparks creative inspiration but also ensures layout versatility. Iris mitigates the workload from scratch, enabling designers to invest more in the essence of design itself.

2. Broad applicability. Iris is suitable for a variety of design tasks including but not limited to poster layout generation, magazine layout generation, and document image layout generation.

3. Designer-centric interactive design. Iris offers an integrated solution from design creation to final rendering, working closely with designers to ensure the usability and high customizability of the tool, while fully respecting the designers' autonomy and creative vision.

Additionally, the ability of Iris to edit online not only fulfills the designer's need for a directly usable system, but also fills in the gaps that other systems miss, making design easier and faster.

Table 5 Ablation study results on visual backbone in layout generation with intra- and inter-domain constraints

Method	Text readability↓	Product presentation↓
w/o visual backbone	38.34	1.031
w/ ResNet-FPN	<u>30.81</u>	<u>0.772</u>
w/ ViT (ours)	**29.94**	**0.683**

w/o: without; w/: with. The values displayed in bold represent the optimal results, whereas those underlined denote the sub-optimal results. A downward arrow indicates that lower values represent better performance

5.2 System overview

The online UI of the Iris system (as shown in Fig. 5) consists of four main regions: a user input panel, a list of layouts generated, a canvas panel, and an edit panel. Specifically, the user input panel is for the designer to select different layout design tasks and input inter- and intra-domain constraints (e.g., element category labels, element number, scenario, and image), as shown in Fig. 5a. The list of layouts generated comprises the top 10 layouts generated by our model. The canvas panel (Fig. 5c) displays the layout template selected by the designer, and modifications to the layout are made through the edit panel (Fig. 5d).

Fig. 6 provides an overview of the pipeline used by designers when working with Iris. Based on our

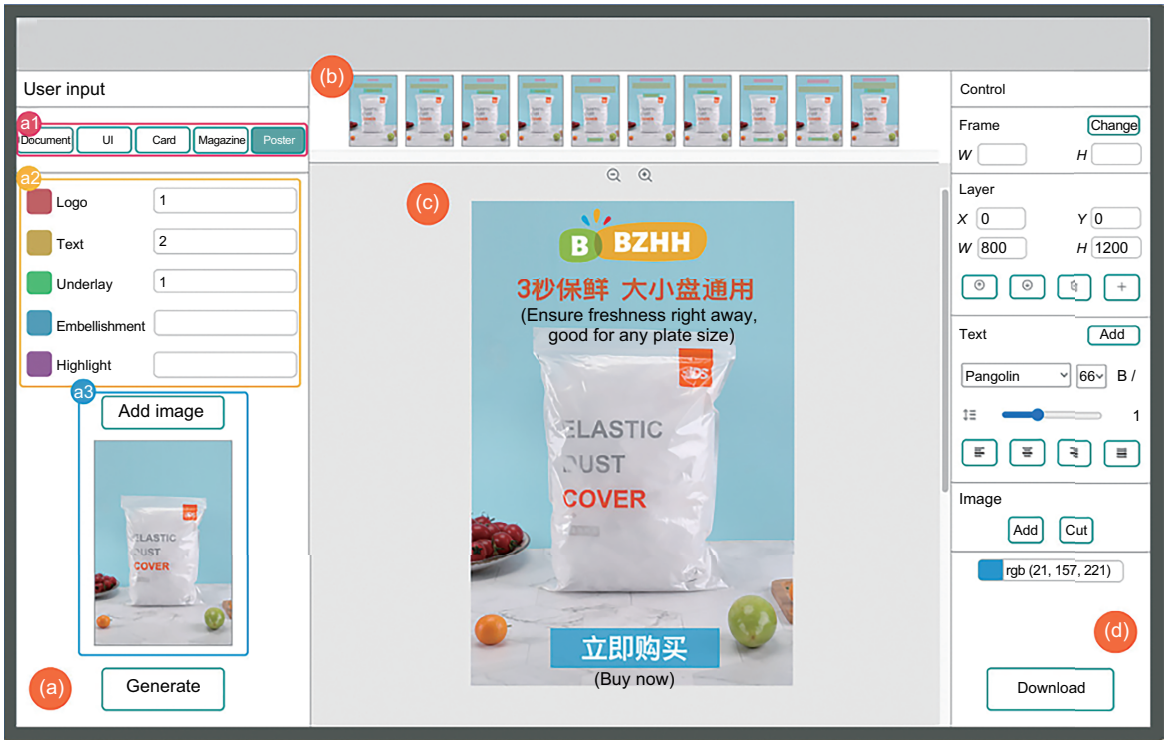

Fig. 5 Iris user interface consisting of a user input panel (a), a list of generated layouts (b), a canvas panel (c), and an edit panel (d)

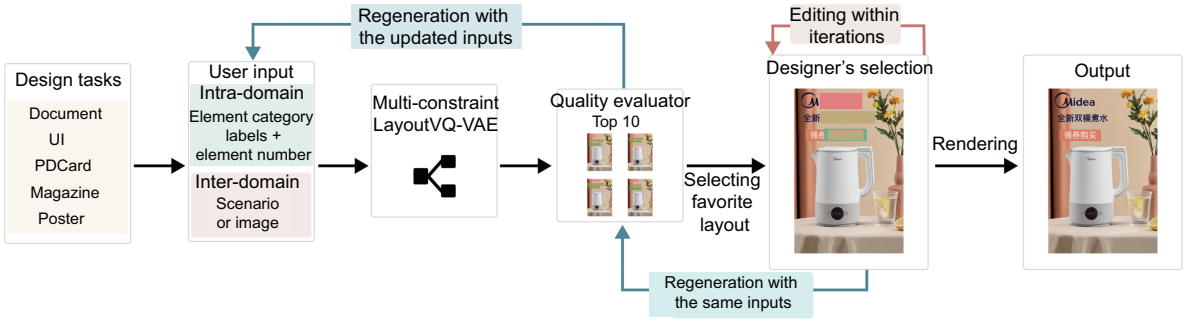

Fig. 6 Overview of the running pipeline

model, we divide design tasks into two main categories: design tasks with only intra-domain constraints (document and UI) and design tasks with intra- and inter-domain constraints (PDCard, magazine, and poster) (G2). For the former design tasks, the designer needs to select the element category and input the required quantities, while for the latter, the designer needs to select the application scenario or import the background image. Then the design materials will be sent to the pre-trained generative model, multi-constraint LayoutVQ-VAE (introduced in Section 3), for automatic layout generation. Within seconds, Iris can generate a large number of high-quality and diversified layouts (G1).

For the generated layouts, considering their large number, a pre-trained layout quality evaluator is employed to rate their quality, based on the overlap and alignment of graphic elements. To inspire designers and ensure diversity in layout design, we preserve the top 10 highest-quality layout designs (G1). If the 10 displayed layouts do not meet the designer's preference, they can be regenerated with the same input. If the designers find the initial inputs unsatisfactory or suboptimal, they can return to the user-input stage, adjust the settings in the user input panel, and operate Iris to regenerate new

layout recommendations. The designers can choose their preferred layout by clicking on it, which will then be shown in the canvas panel. They can further customize the layout by using the functions available in the edit panel to modify the position, size, font, color, and other characteristics of the design elements within the layout. On finalizing these modifications, the design can then be rendered. Through this pipeline, Iris blends the benefits of automated and customized design, breaking down the boundaries between them. It offers a complete design process from generation, editing, to final rendering, ensuring that designers are able to work seamlessly on their designs (G3). Moreover, our system prioritizes user-friendliness, thus making it easy for designers to operate Iris.

5.3 Application scenarios

Iris establishes itself as a novel design tool, catering to multifarious design requirements and pushing the boundaries of conventional layout generation systems. One of its primary domains is document and UI graphic layout design, a realm characterized by the orchestration of multiple elements like text, images, and interactive components to synthesize visually pleasing and functional designs. By modeling the relationship between elements with different categories, Iris swiftly generates large quantities of layouts based on user-specified categories and the number of elements, minimizing the amount of effort and time devoted to trial-and-error compared to a traditional approach. In specialized design tasks such as PDCards (Jing et al., 2023) and magazine layouts, Iris exhibits remarkable adaptability and fluidity in aligning user-defined elements with application scenario constraints. For example, the generated layout applied to a news or science magazine will be more regular and rigorous, and that applied to a fashion or food magazine will be more varied and creative. More generation results can be seen in the supplementary materials. This illustrates Iris's proficiency in integrating layouts with the corresponding scenarios. In the realm of visual–textual presentations, such as posters, Iris incorporates semantic and spatial information of background images, rather than focusing solely on graphic elements. This holistic view enables Iris to produce designs that are both visually captivating and contextually relevant.

Overall, Iris has emerged as a unique solution in the field of design tools, successfully integrating advanced generative algorithms and interactive design environments. It provides continuous, efficient, and intuitive experience during the design generation, editing, and final rendering phases, demonstrating the potential of advanced tools to increase efficiency and innovation in the design field.

5.4 Implementation details

Iris's UI is implemented as a Web application using HTML, CSS, and JavaScript. We developed the system backend in Python. The frontend communicates with the backend using an application programming interface (API) built with Django. The functions of the canvas panel were implemented using knova.js to support the editing functionality of the graphics and the rendering of the generated layouts. Additionally, we used jscolor.js to implement color picking, and loaded fonts from Google Fonts using the Web Font Loader.

6 User study

In this section, we present the results of a user study that was conducted to gain insights and feedback on the potential, limitations, and future opportunities associated with Iris's capability for automatic layout generation. Specifically, our aims are to (1) assess the quality of layouts generated by Iris, (2) assess whether Iris could improve the efficiency of layout design, and (3) gather feedback from designers on the overall usability of Iris.

6.1 Methodology

Our experimentation to verify whether Iris could improve the efficiency associated with the creation of high-quality layouts consisted of two phases: a human–artificial intelligence layout design challenge with 18 participants (phase 1), followed by the layout quality evaluation test taken by 127 participants (phase 2).

6.1.1 Participants

In the first phase, we recruited 18 professional designers who (1) have at least two years of UI design experience (either in the industry or with their own projects) and (2) are skillful in using general graphic design tools (e.g., Sketch and Figma) to

conduct layout design experiments under the given design task. Each designer was paid 10 per hour.

In the second phase, we recruited 45 designers (24 males and 21 females; average age, 25.43 years old; average design experience, 2.76 years) and 82 non-designers (30 males and 52 females; average age, 23.81 years old) through social media platforms to rate the layout quality from the first phase.

6.1.2 Procedure

1. Phase 1

The 18 designers were evenly assigned to two design tasks: the poster design (task 1) and the magazine layout design (task 2). Task 1 was primarily to assess Iris's ability to generate high-quality poster layouts under background image constraints, while task 2 was to evaluate Iris's capacity to create magazine layouts under application scenario constraints.

In task 1, nine designers were evenly divided into three groups (designer, Luban, and Iris groups). Each designer was asked to design three product posters based on three given product images and several corresponding design elements. In particular, the designer group was asked to use general graphic design tools (Figma) to create posters, while the Luban group was instructed to use Luban (https://luban.aliyun.com/), an AI design system, to generate posters, and the Iris group was instructed to use our proposed system, Iris. We provided the three groups with the same materials (i.e., visual elements in the scalable vector graphic (SVG) format and textual information). In task 2, the other nine designers were asked to create a magazine layout based on the content of a given magazine, and they were evenly divided into three groups (designer, template, and Iris groups). In particular, the template group was required to use the 30 magazine templates provided by the magazine dataset. The settings of the designer and Iris groups were the same as those involved in task 1. Each designer was asked to design three different magazine layouts, one for each magazine category (i.e., news, fashion, and travel magazines).

Before the experiment, we conducted a demographic survey for each designer and then provided them with a detailed introduction to the experiment to help them understand the purpose of our study and the specific tasks they need to complete. Additionally, we provided supplementary design materials to help designers become acquainted with the design task and familiarize themselves with the corresponding systems. Once the designers confirmed their familiarity with either Luban or Iris, they moved forward with the assigned task. All designers were required to submit their designs within 1 h. At the end of the experiment, for task 1, 27 posters designed by the three groups (designer, Luban, and Iris) were collected (some of them are shown in Fig. 7, task 1); for task 2, 27 magazines designed by another three groups (designer, template, and Iris) were collected (some of them are shown in Fig. 7, task 2).

Finally, we asked participants to respond to the system usability scale (SUS) questionnaire (Bangor et al., 2009) and the NASA task load index (NASA-TLX) questionnaire (Hart and Staveland, 1988). Thereafter, we engaged in in-depth interviews with designers to discuss the usability of Iris. Throughout these discussions, the designers shared open-ended feedback on their experience with Iris. By gathering these insights, we gained a better understanding of how to further refine and develop Iris to cater to the diverse needs of users.

2. Phase 2

We recruited 45 designers and 82 non-designers to rate the layout quality from the first phase. Specifically, for designers, we randomly selected nine posters among the abovementioned 27 posters (task 1) and nine magazine layouts among the abovementioned 27 magazine layouts (task 2). The layouts were shown to the designers one by one in a random order. Designers were tasked with selecting the layout with the highest aesthetic quality from each comparison group (each group consisted of three posters created variously by designer, Luban, or Iris). For each poster layout, designers were asked to evaluate it based on three key aspects: coordination of the overall design, satisfaction with visual representation, and clarity of conveyed text information. Similarly, each magazine layout was evaluated for coordination of the overall design, neatness of alignment of layout elements, and applicability to the scenario. Evaluation was conducted with a five-point Likert scale, ranging from "very poor" to "very good." For non-designers, they were asked to simply choose the layout they perceived to be of the highest quality from each comparison group. Finally, 127 valid questionnaires were processed.

6.2 Results

Fig. 8 illustrates the average evaluation scores of the poster layouts and the magazine layouts. It can be seen that the poster layouts created by the Iris group exhibited a resemblance to those of the designer group in terms of coordination, poster presentation, and text clarity, and were superior to those of the Luban group, especially in the first two indicators. In terms of the results concerned with coordination, it was demonstrated that the layout generated by Iris can match the layout created by

Fig. 7 Examples for the posters and magazines designed by the designer, Luban/template, and Iris groups in phase 1

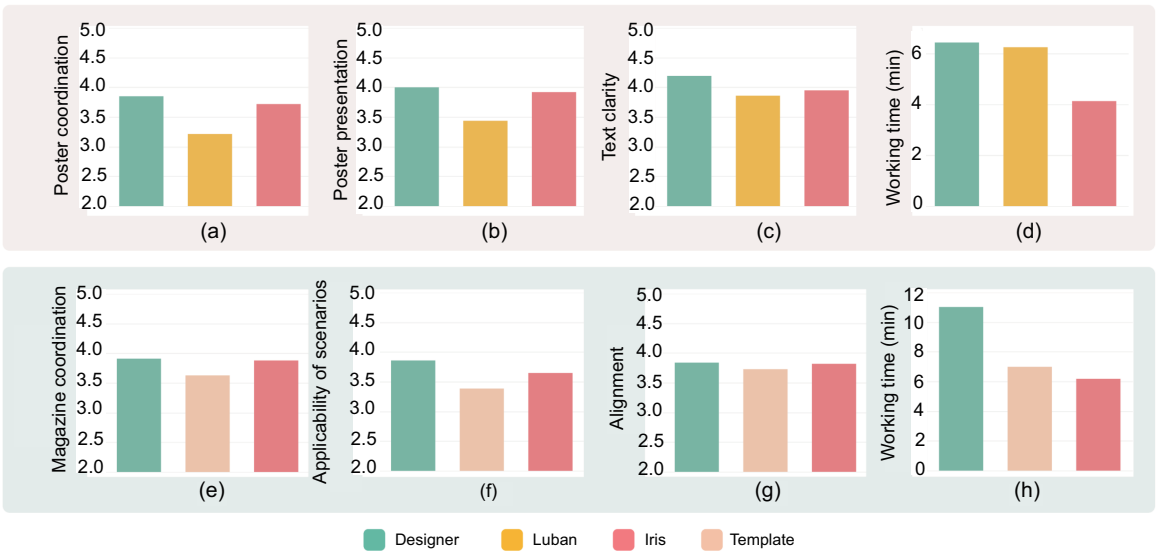

Fig. 8 Results of phase 2 by designers: (a) average rating for poster coordination using a five-point Likert scale; (b) average rating for satisfaction with the visual presentation of the poster; (c) average rating for clarity of conveyed text information on the poster; (d) average time to create a poster; (e) average rating for magazine layout coordination; (f) average matching rate between magazine layout and target scenarios; (g) average rating for alignment of layout elements in magazines; (h) average time to create a magazine layout

designers. The visual presentation of posters reflects whether element labels (such as textual information) obscure product images. The results on this indicator demonstrate that Iris effectively addresses the poster layout generation problem by producing high-quality visual–textual presentation designs for the given images.

The results for the magazine layout task are similar to those for the poster layout; i.e., the Iris group can achieve results comparable to the designer group, and there is a noticeable gap between the performance of the template group and those of the other two groups. Specifically, concerning the applicability of scenarios (Fig. 8f), it is difficult for designers to choose a layout that perfectly fits the scenario requirements and element labels within the limited set of templates, resulting in a relatively low matching accuracy rate. Iris, on the other hand, learns the characteristics of magazine layouts applied to different scenarios and is able to generate varied layouts suitable for the target scenarios.

In terms of design efficiency, Figs. 8d and 8h illustrate the average time spent by the three groups of designers in creating a poster layout and a magazine layout, respectively. It can be seen that both our proposed approach and the template-based approach can significantly reduce the work time of designers. Specifically, in task 1 (poster design), the Iris group spent 59.29% less time on designing a layout for a poster than the designer group. In task 2 (magazine design), the Iris group spent 73.30% less time on designing a layout for a magazine than the designer group. Thanks to Iris ability to generate layouts quickly, designers can design a layout in an average of 4 min, including the optimization of element size, position, layout generation, and element aspect ratio.

Overall, using Iris for layout creation can significantly reduce work time while ensuring the quality of layouts in terms of scenario constraint satisfaction, image constraint satisfaction, aesthetics, and diversity.

6.3 Expert interview

To further evaluate our system, we asked designers to respond to SUS and NASA-TLX, and conducted a semi-structured interview with them. Based on SUS score, Iris demonstrated good usability. The average SUS score of 71.25 (σ =9.45) was above the mean of the Bangor distribution (Bangor et al., 2009) (70.1, with a 99.9% confidence interval in the range of 68.7–71.5), which indicated that users found our system to be user-friendly and effective. The average usability score of 70.31 (σ = 9.63) further supported this conclusion, as it also fell within the range of good usability. Additionally, the high learnability score of 75 (σ = 13.69) suggested that users were able to quickly and easily understand how to use Iris, which is a critical aspect of overall usability.

Fig. 9 shows the average NASA-TLX scores for the designer and Iris groups. The following conclusions can be drawn: the Iris group had a lower score in terms of temporal demand compared to the designer group, with respective scores of 29.17 (σ = 4.49) and 36.25 (σ = 23.94). This indicated that participants experienced less time pressure in using Iris. Furthermore, the performance of the Iris group ($\mu = 35.00$, $\sigma = 8.17$) was slightly better than that of the designer group ($\mu = 40.00$, $\sigma = 9.13$), indicating that participants performed better in the Iris group. These findings suggested that Iris has advantages in terms of improving work efficiency and saving time. Compared to the general graphic design system, Iris allowed participants to complete the task in a shorter time while maintaining high-quality work. Therefore, Iris demonstrated greater efficiency and time management benefits.

Most of the designers believed that Iris was an effective and practical layout generation system. Its

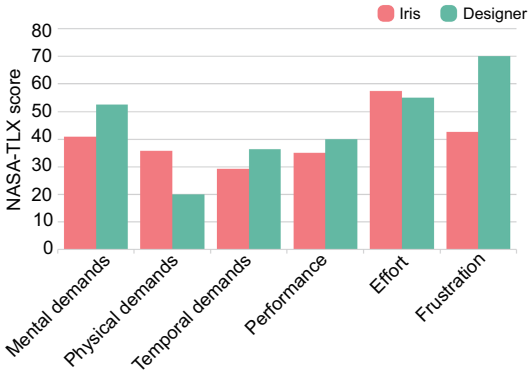

Fig. 9 Average of NASA-TLX measures for the designer and Iris groups. The higher the score for each dimension, the greater the perceived workload or difficulty the user experiences in that aspect, except the performance dimension, in relation to which the scoring is reversed, with higher scores actually representing lower performance

user-friendly interface adheres to the principles of "fast cognition" and "efficient design," simplifying the understanding of the system. Even novices can quickly grasp its functionality and begin creating layouts. Designers expressed their willingness to use Iris for poster generation due to its ability to quickly produce numerous results, allowing them to browse and select their preferred option. Designer D4 appreciated the online editing function, stating that while AI generated appealing posters, they still preferred to fine-tune the design themselves.

Although Iris has gained recognition as a powerful and effective layout generation system, there are opportunities for further improvement. D5 recommended incorporating extra design features, such as reference lines and color balance adjustments, which would transform Iris into a more comprehensive design solution. Furthermore, D3 proposed that Iris should enable users to directly import materials and replace them with a single click. This enhancement would simplify the design process and elevate the overall user experience.

In conclusion, Iris is a promising layout generation system with many advantages, including effectiveness, user-friendliness, and time-saving capabilities. With the feedback from the designers and pursuant further improvement, Iris can further enhance its offerings and cater to an even wider range of user requirements.

7 Conclusions

In the present study, we propose a novel generative model, the multi-constraint LayoutVQ-VAE, to address the challenge of layout generation with both intra-domain constraint (element categories) and inter-domain constraint (background image). Our model employs a VQ-VAE architecture to learn the discrete latent representation of layouts, serializes the element categories and background image, and adopts a Transformer to model the relationship between the constraints and layout representation. We quantitatively and qualitatively evaluate our model on three publicly available datasets and compare it with several state-of-the-art methods. The results demonstrate that our model outperforms or is on par with previous methods, and that it is capable of producing high-quality layouts characterized by excellence in terms of visual performance and aesthetics.

Additionally, to address the high demand for graphic design and the heavy workload of designers in the real world, we introduce Iris, an intelligent layout generation system based on our proposed model. Users can input background images and desired design elements, and the system generates multiple layout designs that fit the input within seconds, allowing for user customization. Iris can be applied to various design scenarios, such as poster design, magazine page design, and mobile UI design. User studies in two different design scenarios demonstrate that, in terms of the quality of layouts generated, our model's output features good comparability with human-designed layouts, significantly reducing the workload and improving design efficiency. In future research, we hope to integrate the content generation ability of image generation models with the layout generation ability of our model to achieve even more intelligent automation in graphic design.

Contributors

Liuqing CHEN, Qianzhi JING, and Yixin TSANG designed the research. Qianzhi JING and Yixin TSANG processed the data. Liuqing CHEN, Qianzhi JING, and Yixin TSANG drafted the paper. Liuqing CHEN and Tingting ZHOU revised and finalized the paper.

Conflict of interest

All the authors declare that they have no conflict of interest.

Data availability

The data that support the findings of this study are available from the corresponding author upon reasonable request.

References

Arroyo DM, Postels J, Tombari F, 2021. Variational Transformer networks for layout generation. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.13637-13647. https://doi.org/10.1109/CVPR46437.2021.01343

Ba JL, Kiros JR, Hinton GE, 2016. Layer normalization. https://arxiv.org/abs/1607.06450

Bangor A, Kortum P, Miller J, 2009. Determining what individual SUS scores mean: adding an adjective rating scale. *J Usabil Stud*, 4(3):114-123.

Cao YN, Ma Y, Zhou M, et al., 2022. Geometry aligned variational Transformer for image-conditioned layout generation. Proc 30th ACM Int Conf on Multimedia, p.1561-1571. https://doi.org/10.1145/3503161.3548332

Dayama NR, Todi K, Saarelainen T, et al., 2020. GRIDS: interactive layout design with integer programming. Proc CHI Conf on Human Factors in Computing Systems, p.1-13. https://doi.org/10.1145/3313831.3376553

Deka B, Huang ZF, Franzen C, et al., 2017. Rico: a mobile App dataset for building data-driven design applications. Proc 30th Annual ACM Symp on User Interface Software and Technology, p.845-854. https://doi.org/10.1145/3126594.3126651

Devlin J, Chang MW, Lee K, et al., 2019. BERT: pre-training of deep bidirectional Transformers for language understanding. Proc Conf of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, p.4171-4186.

Dosovitskiy A, Beyer L, Kolesnikov A, et al., 2021. An image is worth 16×16 words: Transformers for image recognition at scale. Proc 9th Int Conf on Learning Representations.

Guo SN, Jin ZC, Sun FL, et al., 2021. Vinci: an intelligent graphic design system for generating advertising posters. Proc CHI Conf on Human Factors in Computing Systems, Article 577. https://doi.org/10.1145/3411764.3445117

Gupta K, Lazarow J, Achille A, et al., 2021. LayoutTransformer: layout generation and completion with self-attention. Proc IEEE/CVF Int Conf on Computer Vision, p.984-994. https://doi.org/10.1109/ICCV48922.2021.00104

Hart SG, Staveland LE, 1988. Development of NASA-TLX (task load index): results of empirical and theoretical research. *Adv Psychol*, 52:139-183. https://doi.org/10.1016/S0166-4115(08)62386-9

He KM, Zhang XY, Ren SQ, et al., 2016. Deep residual learning for image recognition. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.770-778. https://doi.org/10.1109/CVPR.2016.90

Heusel M, Ramsauer H, Unterthiner T, et al., 2017. GANs trained by a two time-scale update rule converge to a local Nash equilibrium. Proc 30th Int Conf on Neural Information Processing Systems, p.6626-6637.

Hsu H, He XT, Peng YX, et al., 2023. PosterLayout: a new benchmark and approach for content-aware visual-textual presentation layout. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.6018-6026. https://doi.org/10.1109/CVPR52729.2023.00583

Hui MD, Zhang ZZ, Zhang XY, et al., 2023. Unifying layout generation with a decoupled diffusion model. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.1942-1951. https://doi.org/10.1109/CVPR52729.2023.00193

Inoue N, Kikuchi K, Simo-Serra E, et al., 2023. LayoutDM: discrete diffusion model for controllable layout generation. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.10167-10176. https://doi.org/10.1109/CVPR52729.2023.00980

Jacobs C, Li W, Schrier E, et al., 2003. Adaptive grid-based document layout. *ACM Trans Graph*, 22(3):838-847. https://doi.org/10.1145/882262.882353

Jiang ZY, Sun SZ, Zhu JH, et al., 2022. Coarse-to-fine generative modeling for graphic layouts. Proc 36th AAAI Conf on Artificial Intelligence, p.1096-1103. https://doi.org/10.1609/aaai.v36i1.19994

Jiang ZY, Guo JQ, Sun SZ, et al., 2023. LayoutFormer++: conditional graphic layout generation via constraint serialization and decoding space restriction. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.18403-18412. https://doi.org/10.1109/CVPR52729.2023.01765

Jing QZ, Zhou TT, Tsang Y, et al., 2023. Layout generation for various scenarios in mobile shopping applications. Proc CHI Conf on Human Factors in Computing Systems, Article 130. https://doi.org/10.1145/3544548.3581446

Kaiser L, Bengio S, Roy A, et al., 2018. Fast decoding in sequence models using discrete latent variables. Proc 35th Int Conf on Machine Learning, p.2395-2404.

Kikuchi K, Simo-Serra E, Otani M, et al., 2021. Constrained graphic layout generation via latent optimization. Proc 29th ACM Int Conf on Multimedia, p.88-96. https://doi.org/10.1145/3474085.3475497

Kong X, Jiang L, Chang HW, et al., 2022. BLT: bidirectional layout transformer for controllable layout generation. Proc 17th European Conf on Computer Vision, p.474-490. https://doi.org/10.1007/978-3-031-19790-1_29

Li JN, Yang JM, Hertzmann A, et al., 2019. LayoutGAN: generating graphic layouts with wireframe discriminators. Proc 7th Int Conf on Learning Representations.

Li JN, Yang JM, Zhang JM, et al., 2021. Attribute-conditioned layout GAN for automatic graphic design. *IEEE Trans Vis Comput Graph*, 27(10):4039-4048. https://doi.org/10.1109/TVCG.2020.2999335

Lin TY, Dollár P, Girshick R, et al., 2017. Feature pyramid networks for object detection. Proc IEEE Conf on Computer Vision and Pattern Recognition, p.2117-2125. https://doi.org/10.1109/CVPR.2017.106

O'Donovan P, Agarwala A, Hertzmann A, 2014. Learning layouts for single-page graphic designs. *IEEE Trans Vis Comput Graph*, 20(8):1200-1213. https://doi.org/10.1109/TVCG.2014.48

Paszke A, Gross S, Massa F, et al., 2019. PyTorch: an imperative style, high-performance deep learning library. Proc 32nd Int Conf on Neural Information Processing Systems, p.8024-8035.

Schrier E, Dontcheva M, Jacobs C, et al., 2008. Adaptive layout for dynamically aggregated documents. Proc 13th Int Conf on Intelligent User Interfaces, p.99-108. https://doi.org/10.1145/1378773.1378787

van den Oord A, Vinyals O, Kavukcuoglu K, 2017. Neural discrete representation learning. Proc 30th Int Conf on Neural Information Processing Systems, p.6306-6315.

Vaswani A, Shazeer N, Parmar N, et al., 2017. Attention is all you need. Proc 30th Int Conf on Neural Information Processing Systems, p.5998-6008.

Xu CC, Zhou M, Ge TZ, et al., 2023. Unsupervised domain adaption with pixel-level discriminator for image-aware layout generation. Proc IEEE/CVF Conf on Computer Vision and Pattern Recognition, p.10114-10123. https://doi.org/10.1109/CVPR52729.2023.00975

You WT, Jiang H, Yang ZY, et al., 2020. Automatic synthesis of advertising images according to a specified style. *Front Inform Technol Electron Eng*, 21(10):1455-1466. https://doi.org/10.1631/FITEE.1900367

Zheng XR, Qiao XT, Cao Y, et al., 2019. Content-aware generative modeling of graphic design layouts. *ACM Trans Graph*, 38(4):133. https://doi.org/10.1145/3306346.3322971

Zhong X, Tang JB, Yepes AJ, 2019. PubLayNet: largest dataset ever for document layout analysis. Proc Int Conf on Document Analysis and Recognition, p.1015-1022. https://doi.org/10.1109/ICDAR.2019.00166

Zhou M, Xu CC, Ma Y, et al., 2022. Composition-aware graphic layout GAN for visual-textual presentation designs. Proc 31st Int Joint Conf on Artificial Intelligence, p.4995-5001.

List of supplementary materials