Unifying Layout Generation with a Decoupled Diffusion Model

Mude Hui[1]* Zhizheng Zhang[2] Xiaoyi Zhang[2] Wenxuan Xie[2] Yuwang Wang[3] Yan Lu[2]
[1]Xi'an Jiaotong University [2]Microsoft Research Asia [3]Tsinghua University
{zhizzhang, xiaoyizhang, wenxie, yanlu}@microsoft.com
theflood@stu.xjtu.edu.cn wang-yuwang@mail.tsinghua.edu.cn

Abstract

Layout generation aims to synthesize realistic graphic scenes consisting of elements with different attributes including category, size, position, and between-element relation. It is a crucial task for reducing the burden on heavy-duty graphic design works for formatted scenes, e.g., publications, documents, and user interfaces (UIs). Diverse application scenarios impose a big challenge in unifying various layout generation subtasks, including conditional and unconditional generation. In this paper, we propose a Layout Diffusion Generative Model (LDGM) to achieve such unification with a single decoupled diffusion model. LDGM views a layout of arbitrary missing or coarse element attributes as an intermediate diffusion status from a completed layout. Since different attributes have their individual semantics and characteristics, we propose to decouple the diffusion processes for them to improve the diversity of training samples and learn the reverse process jointly to exploit global-scope contexts for facilitating generation. As a result, our LDGM can generate layouts either from scratch or conditional on arbitrary available attributes. Extensive qualitative and quantitative experiments demonstrate our proposed LDGM outperforms existing layout generation models in both functionality and performance.

1. Introduction

Layout determines the placements and sizes of primitive elements on a page of formatted scenes (*e.g.*, publications, documents, UIs), which has critical impacts on how viewers understand and interact with the information in this page [13]. Layout generation is an emerging task of synthesizing realistic and attractive graphic scenes with primitive elements of different categories, sizes, positions, and relations. It is of high demands for reducing the burden on heavy-duty graphic design works in diverse application scenarios. Recently, there have been some research works studying unconditional generation [1, 7, 10, 16, 28], conditional generation based on user specified inputs (*e.g.*, element types [12, 13, 15], element types and sizes [13, 16] or element relations [12, 15]), conditional refinement based on coarse attributes [24], and conditional completion based on partially available elements [7], *etc*. However, none of them can cope with all these application scenarios simultaneously. This imposes a big challenge in unifying various layout generation subtasks with a single model, including conditional generation upon various specified attributes and unconditional generation from scratch. Towards this goal, the prior work UniLayout [9] takes a further step by proposing a multi-task framework to handle six subtasks for layout generation with a single model. However, the supported subtasks are pre-defined and could not cover all application scenarios, *e.g.*, conditional generation based on specified element sizes. Besides, it does not take into account the combinational cases of several subtasks, *e.g.*, the case wherein some elements have missing attributes to be generated while the others are with coarse attributes to be refined in the same

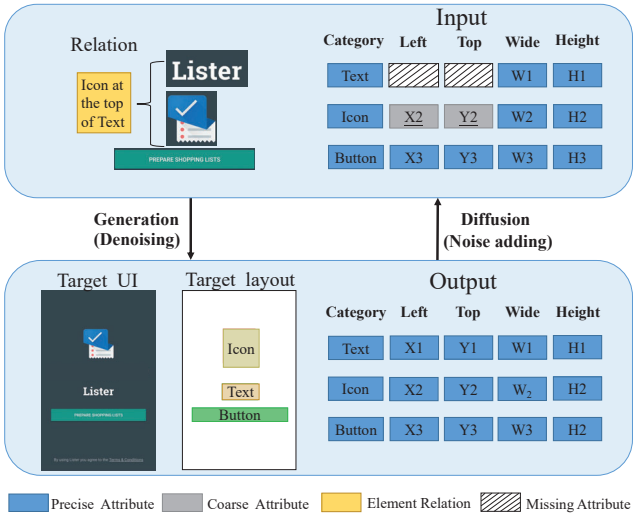

Figure 1. The layout generation tasks can be unified into a diffusion (noise-adding) process and a generation (denoising) process.

*This work was done when Mude Hui was an intern at MSRA.

layout simultaneously.

Generally, a layout comprises a series of elements with multiple attributes, *i.e.*, category, position, size and between-element relation. Each element attribute has three possible statuses: precise, coarse or missing. Different layout generation subtasks supported by previous works are defined as a limited number of cases where the attribute statuses are fixed upon attribute types, as shown in Figure 2. From a unified perspective, all missing or coarse attributes can be viewed as the corrupted results from their corresponding targets. With this key insight in mind, we innovatively propose to unify various forms of user inputs as intermediate statuses of a diffusion (corruption) process while modeling generation as a reverse (denoising) process.

Furthermore, attributes with different corruption degrees are likely to appear at once in user inputs. And different attributes have their own semantics and characteristics. These in fact impose a challenge for the diffusion process to create diverse training samples as comprehensive simulation for various user inputs. In this work, we propose a decoupled diffusion model LDGM to address this challenge. The meaning of "decoupled" here is twofold: *(i)* we design attribute-specific forward diffusion processes upon the attribute types; *(ii)* we decouple the forward diffusion process with the reverse denoising process, wherein the forward processes are individual for different types of attributes, whereas the reverse processes are integrated into one to be jointly performed. In this way, our proposed LDGM includes not only attribute-aware forward diffusion processes for different attributes to ensure the diversity of generation results, but also a joint denoising process with fully message passing over the global-scope elements for improving the generation quality. Our contributions can be summarized in the following:

- We present that various layout generation subtasks can be comprehensively unified with a single diffusion model.
- We propose the Layout Diffusion Generative Model (LDGM), which allows parallel decoupled diffusion processes for different attributes and a joint denoising process for generation with sufficient global message passing and context exploitation. It conforms to the characteristics of layouts and achieves high generation qualities.
- Extensive qualitative and quantitative experiment results demonstrate that our proposed scheme outperforms existing layout generation models in terms of functionality and performance on different benchmark datasets.

2. Related Works

2.1. Layout Generation

Layout generation is a burgeoning research topic of synthesizing graphic scenes upon user requirements, facilitating manual design works in diverse applications. Early works in this area are commonly based on GAN [12, 16, 17, 29] or VAE [10, 11, 15, 22]. Recently, transformer models [1, 7] are emerging in this field to improve the generation diversity and quality. They are still difficult to achieve controllable generation since they predict attributes sequentially. To address this problem, BLT [13] employs a bidirectional transformer to achieve parallel decoding.

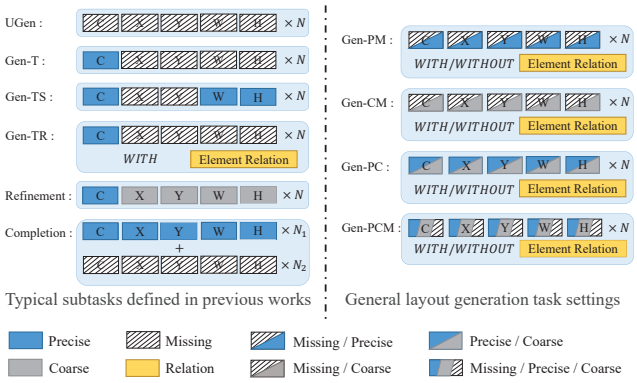

Figure 2. General task settings. The typical layout generation subtasks (left) can be covered by more general task definitions (right).

In this field, the versatility across different generation subtasks is critical to make this technology practical in industry. Towards this goal, multi-task schemes [9, 13] are studied. They are able to handle multiple subtasks simultaneously, but are limited to these pre-defined subtasks only. They do not consider the deep connection among various subtasks, and are thus unable to cover all task types in practical layout generation applications. In this work, we study a versatile framework giving consideration to both the performance and functionality.

2.2. Diffusion Models

Diffusion generative models [8, 20, 31] have recently emerged as a new class of generative models of high performance. They use variational inference to produce training samples by adding noises until the signal is corrupted corresponding to a forward diffusion process, and learns to generate the signal through multi-step denoising corresponding to a reverse denoising process. It is firstly proposed by Sohl-Dickstein *et al*. [26] and regains widespread attention due to its rather impressive performance in generating images [2, 6, 8, 20, 27], texts [2, 5, 18], audio [3, 14, 23], and more. In these works, the signal generation process is decomposed into multiple denoising steps where noises are added during training without distinction on different components/attributes of signals. In this work, considering that elements in a layout include attributes that are of different semantics, we propose a decoupled diffusion model for layout generation to decouple these attributes in noise adding strategies. It comprehensively unifies diverse generation subtasks with a single diffusion model.

3. Problem Definition

A layout $\boldsymbol{l}$ consisting of N elements could be represented as a fully connected graph, where the edges denote relations between elements. Each element has five attributes described by (c, x, y, w, h). c stands for the category of each element such as the text, image, button, *etc*. (x, y) are the coordinates of the left-top corner of each element bounding box, denoting the information of location. (w, h) describe the element size, corresponding to width and height, respectively. We denote pairwise relations between elements as a matrix $\mathcal{E} \in \mathbb{R}^{N\times N}$. As a result, such a layout could be formulated as $\boldsymbol{l} = [c_1, x_1, y_1, w_1, h_1, c_2, x_2, y_2, \cdots, h_N, \mathcal{E}]$.

Layout generation aims to predict all variables in $\boldsymbol{l}$ upon user requirements. For conditional layout generation, only partial attribute variables are available as conditions to generate the others. Unconditional layout generation requires the generation of all variables in $\boldsymbol{l}$ with only their total number N given. Existing multi-task layout generators [9, 13] split the attribute variables in $\boldsymbol{l}$ into conditions and the ones to be predicted with rather limited number of protocols (3 in [13] while 6 in [9]) according to the types of predefined subtasks. In this work, we make the first endeavour to eliminate this limitation towards comprehensive versatility.

4. Layout Diffusion Generative Model

Our goal is to design a versatile framework that allows to take arbitrary attribute variables in $\boldsymbol{l}$ as conditions to predict the missing ones or refine the coarse ones. All elements after generation shall be able to be composed into a graphic layout that is functional and aesthetically pleasing. A big challenge for this lies in unifying multiple generation subtasks with a single model. Our key insight for addressing this is that *the process from a completed layout to fully corruption can be modeled as a diffusion process, wherein partially available attribute variables in $\boldsymbol{l}$ can be viewed as corrupted results of the corresponding targets.*

With the above key insight, we propose Layout Diffusion Generative Model (LDGM). Similar with prior diffusion generative models, LDGM decomposes a generation process into successive denoising steps from noisy signals. It destroys training samples by successively adding noises to them, and then learns to recover them by reversing the noise addition process. Considering the characteristic of layout that different attributes have their own semantics in LDGM, we innovatively propose decoupled diffusion processes with an attribute-specific noise-adding strategy and a joint reverse denoising process.

4.1. Unification with Diffusion Modeling

We pinpoint that a missing or coarse attribute in layouts could be viewed as the corrupted result of a complete one through a forward Markov diffusion process. In this section, we first give a unified formulation of the diffusion attributes and then elaborate our proposed decoupled corruption (noise-adding) strategy for different attributes.

For problem simplification, we quantize geometric attributes x, y, w, h as integers following the common practices [9, 13] in this field. So, attributes in $\boldsymbol{l}$ are all discrete variables. Like VQ-diffusion [6], we model a discrete diffusion process with a status transition matrix. Given an attribute $x \in \{1, 2, \cdots, K\}$ at time $t-1$, denoted by x_{t-1}, the probabilities that x_{t-1} transits to x_t could be represented by the matrix $[Q_t]_{ij} = q(x_t = i|x_{t-1} = j) \in \mathbb{R}^{K\times K}$. The forward Markov diffusion process can be formulated as:

$$q(x_t|x_{t-1}) = \mathbf{x}_t^\top Q_t \mathbf{x}_{t-1}, \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{K\times 1}$ is the corresponded one-hot vector of x. According to the property of Markov chains, the probability of x_t from x_0 can be directly marginalized out as:

$$q(x_t|x_0) = \mathbf{x}_t^\top \overline{Q}_t \mathbf{x}_0, \text{where } \overline{Q}_t = Q_1 Q_2 \cdots Q_t. \tag{2}$$

Conditioned on x_0, we can infer the posterior of this diffusion process by:

$$\begin{aligned} q(x_{t-1}|x_t, x_0) &= \frac{q(x_t|x_{t-1}, x_0)q(x_{t-1}|x_0)}{q(x_t|x_0)} \\ &= \frac{\left(\mathbf{x}_t^\top Q_t \mathbf{x}_{t-1}\right)\left(\mathbf{x}_{t-1}^\top \overline{Q}_{t-1} \mathbf{x}_0\right)}{\mathbf{x}_t^\top \overline{Q}_t \mathbf{x}_0}. \end{aligned} \tag{3}$$

Decoupled corruption (noise-adding) strategy. Layout is a graphic representation whose different attributes have their own semantics. A unified framework of the comprehensive versatility requires that the model can condition on any available precise attributes to generate or refine the remaining ones. To ensure the diversity of training samples, as shown in Algorithm 1, we propose to decouple the entire forward diffusion process into three with their individual timelines for the attributes of category c, position (x, y) and size (w, h) and corrupt these three groups with different noises. Considering they are all discrete variables, similar to [6], we adopt mask-and-replace strategies for their diffusion processes, which has a unified formulation as:

$$Q_t = \begin{bmatrix} \alpha_t & \beta_t & \beta_t & \cdots & 0 \\ \beta_t & \alpha_t & \beta_t & \cdots & 0 \\ \beta_t & \beta_t & \alpha_t & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \gamma_t & \gamma_t & \gamma_t & \cdots & 1 \end{bmatrix}, \tag{4}$$

where $\alpha_t, \beta_t, \gamma_t \in [0, 1]$. In the diffusion process, the probabilities for each attribute variable to remain its original value, to be replaced with another value, and to be masked to be an absorbing status at the current time step are α_t, β_t, and γ_t, respectively. Such absorbing status is easy to be

identified by networks and appears more as time t increases, playing the role of embedding the temporal information about t in each decoupled diffusion process. With this unified formulation, we adopt different instantiations for category attribute c and geometry-related attributes x, y, w, h.

For category c, we adopt noises of a uniform distribution for its diffusion, corresponding to a transition matrix Q_t^c. In Q_t^c, $\alpha_t^c, \beta_t^c, \gamma_t^c$ are all constants for a given t and satisfy that $\alpha_t^c + (K^c - 1)\beta_t^c(1 - \gamma_t^c) + \gamma_t^c = 1$. The β_t^c and γ_t^c increase linearly as time t increases. Here, K^c is the number of element categories in layouts, which varies for different datasets. (Detailed introduction is in the supplementary.)

For position (x, y) and size (w, h), we adopt discretized Gaussian noises [2] for their diffusion processes. Here, we introduce the formulation of their transition matrices, with the one for h as an example. Other attribute variables follow the same formation with different value ranges. For h, the adopted discretized Gaussian noises correspond to a transition matrix Q_t^h wherein γ_t^h is a scalar that increases linearly as time t increases, and α_t^h and β_t^h at the position (i, j) are:

$$[\alpha_t^h]_{ij} = 1 - \textstyle\sum_{j=0, j\neq i}^{K^h} [Q_t^h]_{ij}, \tag{5}$$

$$[\beta_t^h]_{ij} = \frac{(1-\gamma_t^h)\exp\left(-\frac{4|i-j|^2}{(K^h-1)^2\sigma_t^h}\right)}{\sum_{n=-(K^h-1)}^{K^h-1}\exp\left(-\frac{4n^2}{(K^h-1)^2\sigma_t^h}\right)}, \tag{6}$$

where K^h is the number of values for h. And σ_t^h is a linearly increasing hyper-parameter as time t increases, which influences but is not equal to the variance of the discretized Gaussian noises.

4.2. Generation with a Joint Denoising Process

Similar with other diffusion generative models [2, 6, 8], we train a denoising model as the generator to reverse the diffusion processes. In notable contrast to them, as presented in Algorithm 1, our diffusion processes are decoupled for enhancing sample diversities, in which different types of attributes do not share a diffusion timeline. Layouts are highly structured representations. Thus, we propose a joint reverse denoising process for generation from scratch or corrupted layouts that consists of attributes with different degrees of corruption. Besides the exploitation of global-scope contexts, a joint reverse process enables the generation conditional on given relations between elements. To achieve these, we design a transformer-based model as shown in Figure 3. We introduce the formulation, model architecture and inference in the following.

Mathematically, the generator $p_\theta(x_{t-1}|x_t, \boldsymbol{g}(x_t))$ learns the reverse denoising process by estimating the transition posterior $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$. The $\boldsymbol{g}(x_t)$ denotes the global-scope contexts of x_t including other attributes in the current layout and the given relations between elements. This model is trained by optimizing the variational upper bound

Algorithm 1 Training of the LDGM

Require: Transition matrices $\{Q_t^c, Q_t^p, Q_t^s\}$, initial network parameters θ, loss weight λ, and learning rate η.

repeat
  $l \leftarrow$ sample a layout from the training set
  $timsteps = \text{zeros}(\text{len}(l))$ ▷ Record t of attributes.
  $\hat{l} = \text{RandSelect}(l)$ ▷ Select attributes for diffusion.
  $\hat{l} = [C, P, S]$ ▷ Group $\hat{l}$ upon the semantics.
  for g in $[C, P, S]$ **do**
    sample $t \sim \text{Uniform}(\{1, \cdots, T\})$
    for x in g **do**
      $timsteps[x.index] = t$
      $x = x_t \leftarrow$ sample from $q(x_t|x_0)$ ▷ Eqn. 2
    end for
  end for
  $\mathcal{L}_x = \begin{cases} \lambda\mathcal{L}_{rec}, & \text{if } timsteps[x.index] = 0 \\ \mathcal{L}_0, & \text{if } timsteps[x.index] = 1 \\ \mathcal{L}_{t-1}, & \text{otherwise} \end{cases}$
  $\mathcal{L} = \sum_{x\in l}\mathcal{L}_x$
  $\theta \leftarrow \theta - \eta\nabla_\theta\mathcal{L}$ ▷ Update network parameters.
until converged

on corrupted (missing or coarse) attributes, *i.e.*, $\mathcal{L}_{vlb} = \mathbb{E}_{q(\mathbf{x}_0)}\left[\sum_{t=0}^{T}\mathcal{L}_t\right]$, and a reconstruction objective $\mathcal{L}_{rec}$ on precise attributes. The $\mathcal{L}_t$ in the variational upper bound can be detailed as:

$$\mathcal{L}_t = \begin{cases} -\log p_\theta(x_0|x_1, \boldsymbol{g}(x_1)), & t=0 \\ D_{kl}(q(x_t|x_{t+1}, x_0)||p_\theta(x_t|x_{t+1}, \boldsymbol{g}(x_{t+1}))), & t\in[1,T) \\ D_{kl}(q(x_T|x_0)||p(x_T)), & t=T \end{cases} \tag{7}$$

where T is the maximum timestep in the diffusion process. Note that $p(\boldsymbol{x}_T)$ is the prior noise distribution that can be computed in advance during training. For the precise attributes without corruption, we adopt a reconstruction loss as below to ensure their preservation:

$$\mathcal{L}_{rec} = -\log p_\theta(\hat{x}|x, \boldsymbol{g}(x)), \tag{8}$$

Where $\hat{x}$ refers to the output of our generative model for x. The overall loss is a weighted sum of $\mathcal{L}_{vlb}$ and $\lambda\mathcal{L}_{rec}$ with a hyperparameter λ:

$$\mathcal{L} = \mathcal{L}_{vlb} + \lambda\mathcal{L}_{rec}. \tag{9}$$

Model architecture. In LDGM, as illustrated in Figure 3, we adopt a transformer-based model to implement $p_\theta(x_{t-1}|x_t, \boldsymbol{g}(x_t))$. We tokenize each attribute of elements in the layout $\boldsymbol{l}$ with its relevant information including the *value*, *type*, *position embedding* and *condition flag*. Here, the type is an index to identify the category of this attribute while the position embedding is the embedding of element-level indexes indicating which layout element this attribute

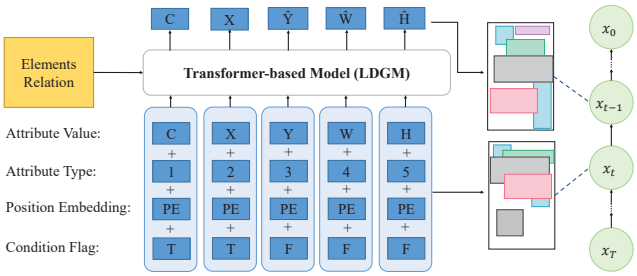

Figure 3. Overall framework of our method. The input attributes of different subtasks can be considered as different x_t. LDGM gradually denoise them to x_0 as the final generation results.

belongs to. The condition flag is a binary scalar to tell this attribute is precise or corrupted. All information is vectorized and then fused into one attribute token by a summation operation. Tokens in $\boldsymbol{l}$ are taken as the inputs of the generator p_θ to infer the denoising result for each attribute, in which a global-scope message passing over all layout elements and their attributes is performed via self-attention.

As introduced in Section 3, LDGM supports the generation conditional on given relations. Given N elements in layout $\boldsymbol{l}$, the pairwise relations can be represented by a matrix $\mathcal{E} \in \mathcal{R}^{N\times N}$. Each element in this matrix has nine possible discrete values, including three on size (*i.e.*, *smaller*, *larger*, and *equal*), five on location (*i.e.*, *above*, *bottom*, *left*, *right*, and *overlapped*) and another one to denote "*unavailable*". We embed each value to be two vectors of the same dimension with the input token with two different embedding layers for query tokens and key tokens, respectively, yielding $V_r^K, V_r^Q \in \mathbb{R}^{N\times N\times d}$. We integrate such relation information into the generation process via relative position embedding proposed in [25], formulated by:

$$e_{i,j} = \frac{(\boldsymbol{x}_i W^Q + [V_r^Q]_{i,j})(\boldsymbol{x}_j W^K + [V_r^K]_{i,j})}{\sqrt{d}}, \quad (10)$$

where i and j are token indexes, $\boldsymbol{x} \in \mathbb{R}^{1\times d}$ is the vector of x. The final attention weight between these two tokens is:

$$a_{i,j} = exp(e_{i,j}) / \textstyle\sum_{k=1}^{N} exp(e_{i,k}). \quad (11)$$

Model inference. We propose a confidence-based inference strategy for LDGM in Algorithm 2, which can prevent generation errors from spreading across tokens via transformer in successive denoising steps. For missing attributes, their corresponding probabilities can be taken as confidence scores. At each denoising step, we merely preserve the predicted results of missing attributes with top-k high confidences and re-mark the remaining ones as absorbing status until all missing attributes are predicted. This operation is denoted by Top-kKeep$(\cdot)$ in Algorithm 2 for brevity. For coarse attributes, they are continuously refined until the end of denoising. More details are in Algorithm 2.

Algorithm 2 Inference of the LDGM

Require: Initial layout $\boldsymbol{l}_T$, condition flags, and maximum denoising steps T.

$\boldsymbol{l}_T \leftarrow$ tokenize $\boldsymbol{l}_T$ with condition flags
$\boldsymbol{l}_T^m \leftarrow$ GetMiss$(\boldsymbol{l}_T)$ ▷ Get missing attributes from $\boldsymbol{l}_T$.
$N_m \leftarrow len(\boldsymbol{l}_T^m)$
$k \leftarrow \lceil N_m / T \rceil$
for $t = T, \cdots, 1$ **do**
  $p_\theta(\boldsymbol{l}_{t-1}|\boldsymbol{l}_t) = LDGM(\boldsymbol{l}_t)$
  $\boldsymbol{l}_{t-1}, \boldsymbol{p}_{t-1} \leftarrow$ sample from $p_\theta(\boldsymbol{l}_{t-1}|\boldsymbol{l}_t)$
  if $N_m > 0$ **then**
    $\boldsymbol{l}_{t-1}^m, \boldsymbol{p}_{t-1}^m \leftarrow$ GetMiss$(\boldsymbol{l}_{t-1}, \boldsymbol{p}_{t-1})$
    $\boldsymbol{l}_{t-1}^m \leftarrow$ Top-kKeep$(\boldsymbol{l}_{t-1}^m, \boldsymbol{p}_{t-1}^m)$
    $N_m \leftarrow N_m - k$
  end if
end for
return $\boldsymbol{l}_0$

5. Experiments

5.1. Experiment Setup

Datasets. We conduct ablation and comparison experiments on three public datasets, *i.e.*, Magazine [29], Rico [4] and PubLayNet [30]. Magazine [29] is a dataset of magazine pages with 6 layout element categories, containing 4K+ images. Rico [4] contains 66K+ images of UIs for mobile applications with 27 element categories. PubLayNet [30] comprises 360K+ machine annotated document images with 5 element categories. Following the common practices in previous studies [12, 15, 16], we clean the datasets to improve the quality of the datasets. For Rico dataset, only 13 most frequent categories are remained and the elements out of these categories are removed from the dataset. For both Rico and PubLayNet datasets, we remove the samples containing more than 25 elements. Since the splitting protocols for training and testing are not consistent over different publications, we re-implement their proposed methods and report evaluation results with the same data splitting protocol for fair comparison in the following sections. Detailed introduction for datasets and their configurations can be found in our supplementary.

Evaluation metrics. We adopt four widely-used evaluation metrics (↑: the bigger the better. ↓: the smaller the better). They are introduced in the following. *Maximum Intersection-over-Union (MaxIoU)* (↑) [12] measures the similarity of the elements in bounding boxes of the same category label between the collections of generated layouts and ground-truth layouts. *Frechet Inception Distance (FID)* (↓) measures the distributional distance between the feature representations of generated layouts and their ground truth. Following [12], we train a model to classify whether the input layout is corrupted or not, and use the output of the

penultimate layer for FID computation. *Alignment* (↓) [17] is used to measure the alignment of elements in generated layouts for aesthetics assessment. We compute this metric with respect to six items: *left border, center at x-axis, right border, top border, center at y-axis, and bottom border*. *Overlap* (↓) [17] measures the overlapping degrees between each element pair inside generated layouts. A well-designed layout typically has less element overlaps.

Evaluation subtasks. We evaluate our LDGM on six existing layout generation subtasks previously defined in [7,9,13,24] for performance comparison. Besides, we unify them into four more general settings and evaluate LDGM on these settings to demonstrate our proposed scheme can provide more comprehensive versatility.

- *Unconditional generation (U-Gen)* aims at generating a layout with no input conditions provided by users.
- *Generation conditioned on types (Gen-T)* is to generate a layout conditioned on specified element types.
- *Generation conditioned on types and sizes (Gen-TS)* generates a layout with specified element types and sizes.
- *Generation conditioned on types and relations (Gen-TR)* generates a layout conditioned on specified element types and pairwise element relations.*
- *Refinement* updates coarse attributes of elements in a layout to be more reasonable and realistic. †
- *Completion* aims at generating the missing attributes in a logout from the given/specified ones.

As we discuss in Section 3, each element in a layout can be described with five attributes, *i.e.*, (c, x, y, h, w). Each attribute has three possible statuses in total: precise (P), coarse (C), or missing (M). When two of these three statuses may appear for any attribute, there are three combined settings, *i.e.*, *Gen-PM*, *Gen-CM*, and *Gen-PC*. When these three statuses are all allowed, it corresponds to a most general setting, *i.e.*, *Gen-PCM*. Note that all subtasks including six previously defined ones, *Gen-PM*, *Gen-CM*, and *Gen-PC* can be viewed as the instantiations of *Gen-PCM*.

Implementation detail. We set both maximum diffusion steps and denoising steps to 100. For the network in LDGM, we use 8 eight-head transformer layers. The embedding dimension is and the feed-forward dimension is 2048. We implement our proposed LDGM with PyTorch [21] and adopt the AdamW optimizer [19] with $\beta_1 = 0.9$ and $\beta_2 = 0.98$ for model training on NVIDIA V100 GPUs. The batch size is set to 128, and the learning rate is 5e-5. Linear warmup schedule is adopted and the warmup proportion is set to 0.1. The λ for loss weighting is set to 0.1. More implementation details (*e.g.*, hyper-parameters) are in our supplementary.

*Like CLG-LO [12], we randomly sample 10% relations as the inputs.

†Following RUITE [24], we synthesize an input layout by turning the precise attributes in real layout into coarse attributes, with the noise sampled from a normal distribution (mean: 0, standard deviation: 0.01).

5.2. Quantitative Results

We compare our proposed LDGM to state-of-the-art (SOTA) methods on the six previously defined subtasks introduced in Section 5.1 to demonstrate its performance superiority. In addition, we also evaluate LDGM on our newly proposed task settings to show the extended functionality towards comprehensive versatility. For fair comparison and convincing evaluation, we generate 1K layouts for each model and report the result averaged over five runs with random seeds for each experiment. The results are shown in Table 1. Their standard deviations are placed in the supplementary.

As can be seen from Table 1, LDGM achieves superior performance to SOTA layout generators across the three public datasets. It suggests that by unifying various layout generation subtasks with diffusion modeling, our proposed LDGM is impressively effective in simultaneously handling multiple layout generation subtasks. Moreover, the quantitative results on newly proposed task settings demonstrate that our LDGM is able to achieve more comprehensive versatility. We thus believe that it can support more diverse user requirements in practical applications.

5.3. Qualitative Comparisons

We compare visualizations of generated layouts from our LDGM and other recent layout generators in Figure 4. It can be observed that LDGM achieves superior generation performance to others with better alignment, less overlaps, and more realistic details. More visualization results, rendered images and analysis can be found in our supplementary.

5.4. Ablation Studies

To validate the effectiveness of our proposed technical components in LDGM, we conduct a series of ablation studies on the most general generation task *Gen-PCM*.

Decoupled corruption strategy. Corruption strategy is crucial for diffusion models [2], which is indiscriminate for different components of signals in previous works while LDGM adopts an attribute-decoupled corruption strategy. We demonstrate the effectiveness of this design by comparing four different diffusion strategies‡: *(i) Non-decoupled strategy*: adding noises for different types of attributes with a shared diffusion timeline. *(ii) Partial-decoupled strategy*: the three types of attributes (*i.e.*, category, size and position) are involved in diffusion processes in order with three individual and partially overlapped diffusion timelines. *(iii) Sequential-decoupled strategy*: adding noises for three types of attributes sequentially with three individual and non-overlapped diffusion timelines. *(iv)* our *Parallel-decoupled strategy (ours)*: adding noises for three types of

‡More details and illustrations are in our supplementary.

Table 1. Experiment results on different layout generation subtasks. Align. denotes the alignment metric.

Subtasks	Methods	Magazine				Rico				PubLayNet			
		MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓	MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓	MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓
U-Gen	LayoutTrans. [7]	0.18	47.84	0.59	47.98	0.46	46.64	0.66	64.10	0.32	49.72	0.37	36.63
	BLT [13]	0.20	44.91	0.55	55.56	0.51	33.81	0.59	67.33	0.34	48.24	0.27	42.79
	UniLayout [9]	0.31	36.61	0.49	**44.50**	**0.62**	26.68	0.40	59.26	0.33	32.29	**0.22**	22.19
	LDGM (Ours)	**0.38**	**32.73**	**0.47**	46.43	**0.62**	**26.06**	**0.36**	**56.35**	**0.46**	**25.94**	0.25	**19.83**
Gen-T	LayoutGAN++ [12]	0.26	36.35	0.54	58.44	0.46	34.43	0.58	59.85	0.36	30.48	0.19	32.80
	BLT [13]	0.22	48.26	0.69	64.01	0.44	39.64	0.57	56.83	0.37	44.86	0.21	38.21
	UniLayout [9]	0.32	28.37	0.51	53.56	0.55	18.06	0.48	57.92	0.41	27.34	0.20	20.98
	LDGM (Ours)	**0.36**	**24.67**	**0.45**	**45.11**	**0.58**	**16.64**	**0.39**	**55.87**	**0.44**	**20.69**	**0.15**	**16.88**
Gen-TS	BLT [13]	0.33	22.72	0.59	61.94	0.51	42.88	0.46	57.74	0.40	24.32	0.16	31.06
	UniLayout [9]	0.35	19.35	0.58	56.43	0.55	20.42	0.49	58.72	0.43	27.47	**0.16**	23.82
	LDGM (Ours)	**0.37**	**17.65**	**0.45**	**44.25**	**0.62**	**12.59**	**0.35**	**55.92**	**0.47**	**19.02**	**0.16**	**10.09**
Gen-TR	CLG-LO [12]	0.27	33.88	0.59	59.43	0.38	38.89	0.54	**56.51**	0.38	31.87	0.21	34.39
	UniLayout [9]	0.36	**19.24**	0.54	49.61	0.57	26.38	0.46	66.93	**0.46**	27.73	0.17	27.35
	LDGM (Ours)	**0.39**	20.58	**0.48**	**47.27**	**0.61**	**16.98**	**0.39**	58.75	0.44	**19.54**	**0.16**	**21.28**
Refinement	RUITE [24]	0.24	44.27	0.64	54.26	0.46	36.70	0.57	64.13	0.32	41.72	0.49	35.74
	UniLayout [9]	0.33	19.78	0.49	49.02	0.56	24.41	0.42	56.04	0.44	22.34	0.11	27.23
	LDGM (Ours)	**0.39**	**14.95**	**0.42**	**37.22**	**0.62**	**13.19**	**0.33**	**52.17**	**0.48**	**15.28**	**0.10**	**13.05**
Completion	LayoutTrans. [7]	0.17	39.36	0.67	55.32	0.46	36.15	0.66	67.10	0.32	41.72	0.37	39.81
	UniLayout [9]	0.23	28.78	0.52	46.43	0.59	25.18	0.45	55.99	0.41	32.04	0.19	22.90
	LDGM (Ours)	**0.38**	**24.35**	**0.49**	**39.26**	**0.60**	**16.42**	**0.36**	**53.15**	**0.44**	**25.31**	**0.10**	**19.45**
Gen-PM	LDGM (Ours)	0.38	27.33	0.47	39.02	0.58	21.64	0.38	56.56	0.46	23.58	0.10	14.11
Gen-CM		0.37	28.74	0.51	43.25	0.57	26.15	0.38	57.74	0.44	24.94	0.11	16.26
Gen-PC		0.37	22.56	0.47	42.95	0.60	18.13	0.36	53.67	0.50	16.42	0.09	12.51
Gen-PCM		0.37	24.45	0.49	44.41	0.59	21.59	0.40	54.77	0.42	25.76	0.14	19.68
GT	-	0.41	9.89	0.43	34.27	0.66	7.05	0.26	49.86	0.64	9.38	0.008	5.18

Table 2. Experiment results on the Rico dataset by varying the corruption strategies on input tokens.

Model	MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓
Non-decoupled	0.56	29.24	0.43	60.04
Partial	0.57	27.71	0.48	**54.24**
Sequential	0.56	26.69	0.43	57.17
Parallel (Ours)	**0.59**	**21.59**	**0.40**	54.77

attributes in parallel with three individual and fully overlapped diffusion timelines. The comparison results are in Table 2. We can observe that our proposed strategy achieves the best *MaxIOU*, *FID* and *Alignment* compared to the other three. It delivers the second best *Overlap* (very close to the best one) since the training samples of this strategy are of the highest diversity thus imposing the largest difficult for model optimization.

Inference strategy. We compare our inference strategy against two common transformer decoding strategies: *(i) Autoregressive*: the tokens are decoded one by one in the sequence order. *(ii) Non-autoregressive*: where tokens are decoded all at once for each time step for T decoding steps in total. As shown in Table 3, *Non-autoregressive* achieves the most inferior performance, which is because it predicts many missing attributes at one time step, leading to propagation of generation errors across tokens. *Autoregressive* requires to decode tokens in a given order, which means it lacks the flexibility to meet user requirements as discussed in [13]. Our proposed strategy LDGM can adaptively decode tokens starting from easier ones based on the confidence scores, enabling the exploitation of more reliable context information when decoding the harder ones.

Table 3. Experiment results on the Rico dataset in terms of different inference strategies.

Model	MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓
AutoReg	**0.60**	23.16	0.42	56.87
Non-AutoReg	0.57	25.14	0.44	58.63
Ours	0.59	**21.59**	**0.40**	**54.77**

Table 4. Ablation study on condition flags. *Retent.* refers to the retention/unchanged ratio of condition attributes free of generative errors. "w/o. C-Flags" denotes discarding condition flags.

Model	MaxIoU ↑	FID ↓	Align. ↓	Overlap ↓	Retent. ↑
w/o. C-Flags	0.55	27.38	0.42	55.54	11.25
Ours	**0.59**	**21.59**	**0.40**	**54.77**	**99.66**

Condition flags. We study the benefits of condition flags by comparing LDGM to the model without them. As presented in Table 4, the use of condition flags not only improves the generation qualities measured by different metrics but also plays an important role in preventing attributes

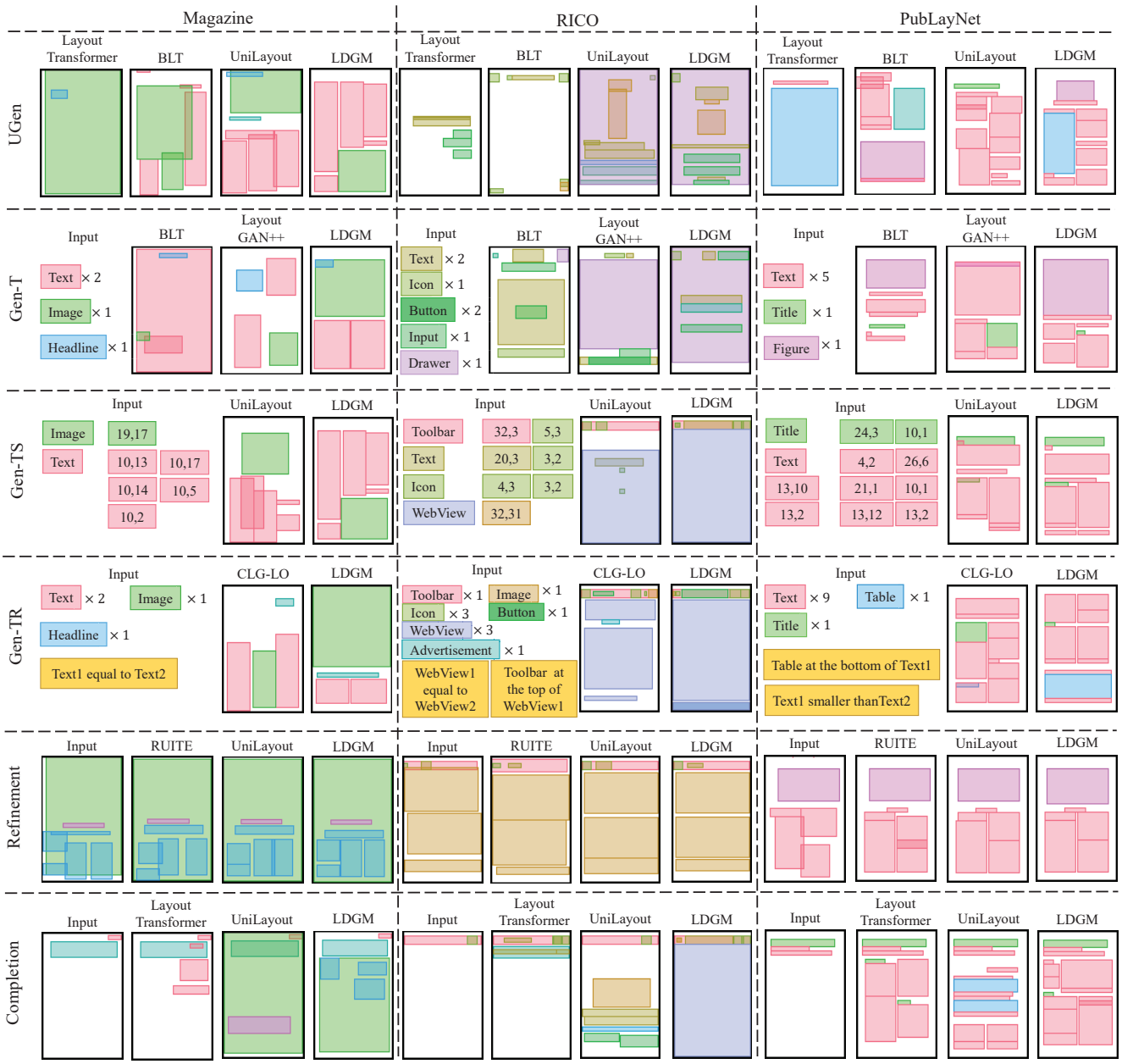

Figure 4. Qualitative comparisons with state-of-the-art layout generation methods.

given as conditions from being destroyed during generation.

6. Conclusion and Future Work

In this work, we unify diverse layout generation subtasks, including unconditional generation from scratch and conditional generation based on various user inputs, with a single diffusion model, *i.e.*, Layout Diffusion Generative Model (LDGM). Given the number of elements in the layout, LDGM supports the generation from arbitrary available element attributes, including category, position, size and relation between elements, no matter they are coarse or precise. To the best of our knowledge, this is the first endeavour to achieve such a comprehensively versatile layout generator. Besides, we devise a decoupled diffusion model which performs decoupled diffusion processes for attributes of different semantics/characteristics and generates them jointly with global-scope contexts taken into account. This methodology presents a core idea of "decouple-first-diffusion-then". We believe this idea will inspire more exploration in designing diffusion-based generative models beyond graphic layouts.

References

[1] Diego Martin Arroyo, Janis Postels, and Federico Tombari. Variational transformer networks for layout generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13642–13652, 2021. 1, 2

[2] Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021. 2, 4, 6

[3] Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. WaveGrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, Sept. 2020. 2

[4] Biplab Deka, Zifeng Huang, Chad Franzen, Joshua Hibschman, Daniel Afergan, Yang Li, Jeffrey Nichols, and Ranjitha Kumar. Rico: A mobile app dataset for building data-driven design applications. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, pages 845–854, 2017. 5

[5] Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022. 2

[6] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022. 2, 3, 4

[7] Kamal Gupta, Justin Lazarow, Alessandro Achille, Larry S Davis, Vijay Mahadevan, and Abhinav Shrivastava. Layouttransformer: Layout generation and completion with self-attention. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1004–1014, 2021. 1, 2, 6, 7

[8] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, pages 6840–6851, 2020. 2, 4

[9] Zhaoyun Jiang, Huayu Deng, Zhongkai Wu, Jiaqi Guo, Shizhao Sun, Vuksan Mijovic, Zijiang Yang, Jian-Guang Lou, and Dongmei Zhang. Unilayout: Taming unified sequence-to-sequence transformers for graphic layout generation. *arXiv preprint arXiv:2208.08037*, 2022. 1, 2, 3, 6, 7

[10] Zhaoyun Jiang, Shizhao Sun, Jihua Zhu, Jian-Guang Lou, and Dongmei Zhang. Coarse-to-fine generative modeling for graphic layouts. In *AAAI Conference on Artificial Intelligence*, 2022. 1, 2

[11] Akash Abdu Jyothi, Thibaut Durand, Jiawei He, Leonid Sigal, and Greg Mori. Layoutvae: Stochastic scene layout generation from a label set. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9895–9904, 2019. 2

[12] Kotaro Kikuchi, Edgar Simo-Serra, Mayu Otani, and Kota Yamaguchi. Constrained graphic layout generation via latent optimization. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 88–96, 2021. 1, 2, 5, 6, 7

[13] Xiang Kong, Lu Jiang, Huiwen Chang, Han Zhang, Yuan Hao, Haifeng Gong, and Irfan Essa. Blt: Bidirectional layout transformer for controllable layout generation. *arXiv preprint arXiv:2112.05112*, 2021. 1, 2, 3, 6, 7

[14] Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020. 2

[15] Hsin-Ying Lee, Lu Jiang, Irfan Essa, Phuong B Le, Haifeng Gong, Ming-Hsuan Yang, and Weilong Yang. Neural design network: Graphic layout generation with constraints. In *European Conference on Computer Vision*, pages 491–506. Springer, 2020. 1, 2, 5

[16] Jianan Li, Jimei Yang, Aaron Hertzmann, Jianming Zhang, and Tingfa Xu. Layoutgan: Generating graphic layouts with wireframe discriminators. *arXiv preprint arXiv:1901.06767*, 2019. 1, 2, 5

[17] Jianan Li, Jimei Yang, Jianming Zhang, Chang Liu, Christina Wang, and Tingfa Xu. Attribute-conditioned layout gan for automatic graphic design. *IEEE Transactions on Visualization and Computer Graphics*, 27(10):4039–4048, 2020. 2, 6

[18] Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *arXiv preprint arXiv:2205.14217*, 2022. 2

[19] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[20] Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. *arXiv preprint arXiv:2102.09672*, 2021. 2

[21] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[22] Akshay Gadi Patil, Omri Ben-Eliezer, Or Perel, and Hadar Averbuch-Elor. Read: Recursive autoencoders for document layout generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 544–545, 2020. 2

[23] Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In *International Conference on Machine Learning*, pages 8599–8608. PMLR, 2021. 2

[24] Soliha Rahman, Vinoth Pandian Sermuga Pandian, and Matthias Jarke. Ruite: Refining ui layout aesthetics using transformer encoder. In *26th International Conference on Intelligent User Interfaces-Companion*, pages 81–83, 2021. 1, 6, 7

[25] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. *NAACL*, 2018. 5

[26] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using

nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265, 2015. 2

[27] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. 2

[28] Kota Yamaguchi. Canvasvae: Learning to generate vector graphic documents. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5481–5489, 2021. 1

[29] Xinru Zheng, Xiaotian Qiao, Ying Cao, and Rynson WH Lau. Content-aware generative modeling of graphic design layouts. *ACM Transactions on Graphics (TOG)*, 38(4):1–15, 2019. 2, 5

[30] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. Publaynet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE, 2019. 5

[31] Ye Zhu, Yu Wu, Kyle Olszewski, Jian Ren, Sergey Tulyakov, and Yan Yan. Discrete contrastive diffusion for cross-modal and conditional generation. *arXiv preprint arXiv:2206.07771*, 2022. 2