Learning Object Placement via Dual-path Graph Completion

Siyuan Zhou, Liu Liu, Li Niu, and Liqing Zhang

MoE Key Lab of Artificial Intelligence, Shanghai Jiao Tong University, China
{ssluvble,Shirlley,ustcnewly}@sjtu.edu.cn, zhang-lq@cs.sjtu.edu.cn

Abstract. Object placement aims to place a foreground object over a background image with a suitable location and size. In this work, we treat object placement as a graph completion problem and propose a novel graph completion module (GCM). The background scene is represented by a graph with multiple nodes at different spatial locations with various receptive fields. The foreground object is encoded as a special node that should be inserted at a reasonable place in this graph. We also design a dual-path framework upon the structure of GCM to fully exploit annotated composite images. With extensive experiments on OPA dataset, our method proves to significantly outperform existing methods in generating plausible object placement without loss of diversity.

1 Introduction

Image composition [2,35,24] refers to the task of producing a realistic composite image based on a background image and a foreground object, which can benefit a wide range of applications of entertainment, virtual reality, and artistic creation. The main concerns of this task include both appearance compatibility (*e.g.*, shading, lighting), geometric compatibility (*e.g.*, object size, camera viewpoint), and semantic compatibility (*e.g.*, semantic context) between foreground and background [2,20]. In this work, we deal with the object placement problem [18,31,38], which is a sub-task of image composition and aims to generate reasonable locations and sizes to place foreground over background. Object placement can be applied in various conditions. For example, during artistic creation, this technique could provide designers with feedback and make recommendation for them when they are placing objects. Another application is automatic advertising, which aims to help advertisers with the product insertion in the background scene [41]. Object placement is a challenging problem, partially due to the lack of annotated composite images. Recently, the first object placement assessment (OPA) dataset [22] was released, which contains composite images and their binary rationality labels indicating whether they are reasonable (positive sample) or not (negative sample) in terms of foreground object placement.

As far as we know, only few works focus on general object placement learning, like TERSE [31] and PlaceNet [38]. Both of them adopt adversarial training to learn the reasonable distribution from real images, and predict a set of parameters

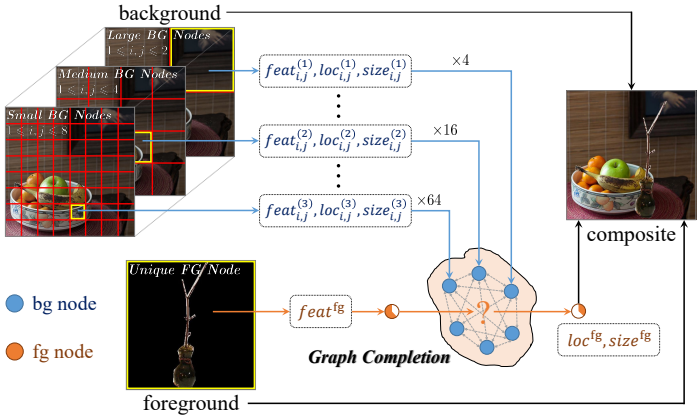

Fig. 1. Illustration of our graph completion module (GCM). Background nodes are extracted from different positions with different receptive fields. The unique foreground node lacks location and size information. GCM infers the missing information of the foreground node to complete the graph

to indicate locations and sizes for placing foreground objects during inference. The drawbacks of these methods mainly come from two aspects. Firstly, they did not explicitly consider the relation between the foreground object and the background scene, which is of great importance in object placement. Secondly, they did not fully exploit the annotated composite images. Based on these considerations, we design our method to generate more plausible and diverse object placements.

To better exploit the relationship between the foreground object and the background scene, we treat object placement learning as a graph completion problem, as shown in Figure 1. On the one hand, the background image can be considered as a graph with multiple nodes. Each node pays attention to a local region on the background feature map with a specific spatial position and receptive field. Multiple background nodes work together to form a graph, enabling the model to discover a variety of plausible solutions on the background. On the other hand, the foreground image can be seen as a special node that should be inserted into the graph with suitable location and size. Note that the background nodes have both feature information and location/size information, whereas the foreground node only has feature information. We need to infer the missing location/size information for the foreground node to complete the graph and obtain a reasonable composite layout.

To complete the graph, we propose a novel graph completion module (GCM) with two components: node extraction head (NEH) and placement seeking network (PSN). NEH aims to transform foreground and background into a node graph. We incorporate one foreground NEH to extract a foreground node and another background NEH to extract multiple background nodes from different positions and scales. PSN contains an attention layer and a regression block. The attention layer attends relevant information from different background regions

for the foreground object and produces an attended feature vector, which will be transmitted to the regression block to predict transformation parameters for object placement. Note that object placement is a multi-modal problem, that is, the reasonable placement has many possible solutions given a pair of foreground and background. This guides us to incorporate a random vector in the regression block to generate diversified transformation parameters for object placement.

To take full advantage of annotated composite images, we design a dual-path framework upon the structure of GCM, including an unsupervised path and a supervised path. The whole framework follows an adversarial learning paradigm, which is composed by a GCM (functioning as a generator) and a discriminator. Transformation parameters produced by GCM are applied to predict reasonable object placements, which are then pushed to the discriminator so as to check the plausibility of the generated composite images. The distinction between two paths lies in the provided data. The unsupervised path only utilizes pairs of foreground and background as input, while the supervised path have additional annotated composite images. Recall that GCM contains a random vector in the regression block. In our implementation, two paths choose different types of random vectors. In the unsupervised path, random vectors are sampled from unit Gaussian distribution. In the supervised path, random vectors (also called latent vectors) are encoded from composite images with positive annotation via a VAE [15] encoder. We expect the generator to reconstruct the ground-truth transformation parameters of each positive composite image from its corresponding latent vector. Under this design, we establish a bijection between the latent vector and the predicted object placement. This can avoid mode collapse [43] and bring multifarious generation results. Since two paths share weights in the generator and the discriminator, we hope that the supervised path could gradually guide the unsupervised path to generate reasonable composite images. During inference, by sampling random vectors in the unsupervised path, we can obtain diverse solutions for object placement. Since the key idea of this work is **Gra**ph **co**mpletion, we name our network **GracoNet**.

In summary, the main contributions of this paper are: 1) We formulate object placement as a graph completion problem and propose a novel graph completion module (GCM). 2) We design a dual-path framework upon GCM to fully exploit annotated composite images and overcome mode collapse issue. 3) Experiments on OPA dataset demonstrate the superiority of our method in generation plausibility and diversity when compared with existing works.

2 Related Work

2.1 Image Composition

The main challenges of image composition [24,29,16,37,17,3] lie in appearance compatibility, geometric compatibility, and semantic compatibility between foreground and background. Up to now, this task has been explored from a variety of perspectives. For example, [42] refined composite images by distinguishing them from natural photographs via a simple CNN model. [14] incorporated scene

graphs to explicitly learn relationships between objects and generate images from a computed scene layout. [2] introduced a new GAN architecture to explore geometric and color correction at the same time. [35] pointed out the drawback of cutting-edge methods and addressed it by a spatially-adaptive mechanism. [36,39] explored the image blending field and achieved seamless connection between foreground and background via blending boundary regions. [21,13] generated realistic shadows *w.r.t* foreground objects over background scenes. [32,6,4,5] proposed image harmonization to deal with color and lighting inconsistency in composite images. Additionally, object placement has been studied to realize geometric compatibility, which will be introduced next.

2.2 Object Placement

Learning object placement has attracted wide attention in recent years. Several early methods [26,10] attempted to design explicit rules to place foreground objects. The followers went a step further to automatically exploit reasonable placement [30,20,18,31,19,38,41,1]. For example, [20] employed spatial transformer networks to learn geometric corrections that warp composite images for appropriate layouts. [18] designed a two-step strategy to find where to place objects and what categories to place. [19] used VAE [15] to predict 3D locations and poses of humans. [1] achieved self-consistency in training a composition network by decomposing composite images back into individual objects. Compared with existing works, we offer a new perspective by treating object placement as a graph completion problem. Our dual-path framework could effectively boost generation plausibility and diversity by discovering placement clues from supervisions.

3 Methodology

Suppose we have a background image $\mathrm{I}^{\mathrm{bg}} \in \mathcal{R}^{3\times H\times W}$ and a foreground image $\mathrm{I}^{\mathrm{fg}} \in \mathcal{R}^{3\times H\times W}$ together with a binary object mask $\mathrm{M}^{\mathrm{fg}} \in \mathcal{R}^{1\times H\times W}$ delineating the foreground object, where H and W represent image height and width. Our objective is to output transformation parameters $\mathbf{t}$ that transform foreground and places it over background to obtain a composite image $\mathrm{I}^{\mathrm{c}} \in \mathcal{R}^{3\times H\times W}$ with a composite foreground mask $\mathrm{M}^{\mathrm{c}} \in \mathcal{R}^{1\times H\times W}$. By using $\mathcal{F}_{\mathbf{t}}$ to represent the transformation function with parameters $\mathbf{t}$, we have

$$(\mathrm{I}^{\mathrm{c}}, \mathrm{M}^{\mathrm{c}}) = \mathcal{F}_{\mathbf{t}}(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}). \tag{1}$$

The detailed definitions of $\mathbf{t}$ and $\mathcal{F}_{\mathbf{t}}$ are left to supplementary. In the following paragraphs, we will first introduce how GCM works to generate transformation parameters $\mathbf{t}$ in Section 3.1. Then, we will introduce the GCM-based dual-path framework that reasonably makes use of supervised information in Section 3.2.

3.1 Graph Completion Module (GCM)

The core module in our network is Graph Completion Module (GCM), which takes in a pair of foreground and background as well as a random vector to

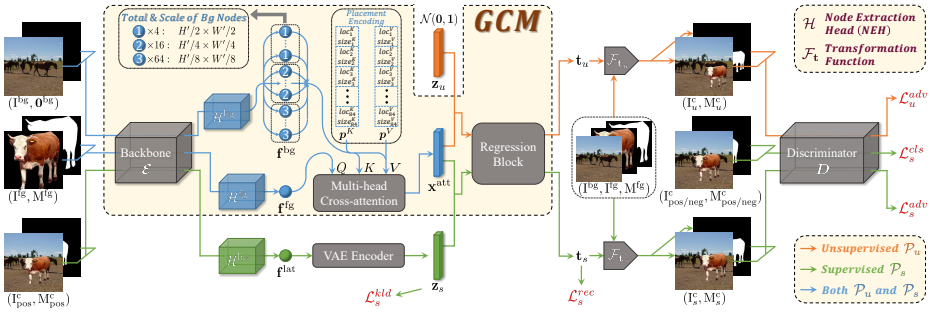

Fig. 2. Our GracoNet has an unsupervised path and a supervised path built upon Graph Completion Module (GCM). GCM consists of backbone network $\mathcal{E}$, node extraction head $\mathcal{H}$, and placement seeking network $\mathcal{S}$ (a multi-head cross-attention layer and a regression block). Loss functions are marked in red. More details are left to Section 3

produce reasonable placement (transformation parameter) for the foreground object. GCM consists of three components: 1) backbone network $\mathcal{E}$, 2) node extraction head $\mathcal{H}$, and 3) placement seeking network $\mathcal{S}$. The backbone network extracts general feature maps for the input images. The node extraction head encodes graph nodes from the feature maps. After that, the placement seeking network finds the relationships between the foreground node and the background nodes, and finally outputs transformation parameters $\mathbf{t}$ that reasonably places the foreground node to complete the graph.

Backbone Network. Our backbone network $\mathcal{E}$ takes in the concatenation of a three-channel image and a one-channel mask to produce a feature map $\mathrm{F} \in \mathcal{R}^{C' \times H' \times W'}$. We use object masks M^{fg} for foreground images I^{fg} and apply all-zero masks $\mathbf{0}^{\mathrm{bg}}$ to background images I^{bg}. Formally, foreground and background features are extracted by $\mathrm{F}^{\mathrm{fg}} = \mathcal{E}(\mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}})$ and $\mathrm{F}^{\mathrm{bg}} = \mathcal{E}(\mathrm{I}^{\mathrm{bg}}, \mathbf{0}^{\mathrm{bg}})$.

Node Extraction Head (NEH). Given a feature map F as input and a positive integer array $\mathbf{n} = [n_1, n_2, \cdots, n_L]$ as parameter, a node extraction head $\mathcal{H}_{\mathbf{n}}$ consists of L node extraction layers named $\mathcal{G}_1, \mathcal{G}_2, \cdots, \mathcal{G}_L$. The l-th layer $\mathcal{G}_l$ evenly divides the feature map F into $n_l \times n_l$ cells and sequentially encodes them into a stack of n_l^2 nodes with dimension C, denoted by $\mathbf{f}^{(l)} \in \mathcal{R}^{n_l^2 \times C}$. Each node accounts for a spatial resolution of $\frac{H'}{n_l} \times \frac{W'}{n_l}$ on the feature map. $\mathcal{H}_{\mathbf{n}}$ gathers the outputs from all L layers, and produces $\mathbf{f} = [\mathbf{f}^{(1)}, \mathbf{f}^{(2)}, \cdots, \mathbf{f}^{(L)}] \in \mathcal{R}^{N \times C}$ with totally $N = \sum_{l=1}^{L} n_l^2$ nodes. Formally, the workflow of $\mathcal{H}_{\mathbf{n}}$ is denoted by $\mathbf{f} = \mathcal{H}(\mathrm{F}; \mathbf{n})$. More implementation details of NEH can be found in Section 4.1.

In GCM, we incorporate a foreground head and a background head, as illustrated in Figure 2. The foreground head $\mathcal{H}^{\mathrm{fg}}_{\mathbf{n}=[1]}$ encodes a global foreground node $\mathbf{f}^{\mathrm{fg}} \in \mathcal{R}^{1 \times C}$ from the foreground feature map F^{fg}, *i.e.*, $\mathbf{f}^{\mathrm{fg}} = \mathcal{H}^{\mathrm{fg}}(\mathrm{F}^{\mathrm{fg}}; [1])$. Meanwhile, the background head $\mathcal{H}^{\mathrm{bg}}_{\mathbf{n}=[2,4,8]}$ produces $N = 84$ local background nodes

$\mathbf{f}^{\text{bg}} \in \mathcal{R}^{84\times C}$ from three scales at different locations, *i.e.*, $\mathbf{f}^{\text{bg}} = \mathcal{H}^{\text{bg}}(\text{F}^{\text{bg}}; [2, 4, 8])$. On the whole, the unique foreground node and all the 84 background nodes work together to form a node graph, as shown in Figure 1.

Placement Seeking Network (PSN). It is noteworthy that the node graph is now incomplete, because the location/size of the foreground node awaits to be determined. To address this issue, we introduce a placement seeking network $\mathcal{S}$, which consists of a multi-head cross-attention layer and a regression block.

First, we use a Transformer multi-head attention layer [33] to explore the relationship between the unique foreground node $\mathbf{f}^{\text{fg}} \in \mathcal{R}^{1\times C}$ and the 84 local background nodes $\mathbf{f}^{\text{bg}} \in \mathcal{R}^{84\times C}$ by treating $\mathbf{f}^{\text{fg}}$ as query and $\mathbf{f}^{\text{bg}}$ as key/value (*i.e.*, cross-attention). Inspired by Transformer that incorporates position encoding [9,33,23,27,7,25] into the attention layer, we introduce placement encoding to encapsulate both location and size information of $\mathbf{f}^{\text{bg}}$. Since different background nodes have distinct positions/scales, they should have different placement encodings. In our implementation, placement encoding includes a learnable $\boldsymbol{p}^K$ (*resp.*, $\boldsymbol{p}^V$) for key (*resp.*, value), which is based on but not exactly the same as the encoding form in [27]. In general, we denote the output of the attention layer by $\mathbf{x}^{\text{att}} = \mathit{Attention}(\mathbf{f}^{\text{fg}}, \mathbf{f}^{\text{bg}}) \in \mathcal{R}^{1\times C}$, and we will introduce the details as follows.

Specifically, we calculate the output $\boldsymbol{o} \in \mathcal{R}^{1\times d_{\boldsymbol{o}}}$ of each attention head:

$$\boldsymbol{o} = \sum_{j=1}^{84} \alpha_j (\mathbf{f}_j^{\text{bg}} W^V + \boldsymbol{p}_j^V), \tag{2}$$

where coefficient α_j represents the edge weight between the foreground node and the j-th background node in the graph:

$$\alpha_j = \text{Softmax}\left(\frac{(\mathbf{f}^{\text{fg}} W^Q)(\mathbf{f}_j^{\text{bg}} W^K + \boldsymbol{p}_j^K)^\top}{\sqrt{d_{\boldsymbol{o}}}}\right). \tag{3}$$

In Eqn.(2) and Eqn.(3), $W^Q, W^K, W^V \in \mathcal{R}^{C\times d_{\boldsymbol{o}}}$ are linear learnable weights for query, key, and value, respectively. $\boldsymbol{p}^K, \boldsymbol{p}^V \in \mathcal{R}^{84\times d_{\boldsymbol{o}}}$ are learnable placement encodings. Following [33], we incorporate 8 attention heads and set $d_{\boldsymbol{o}}$ as $\frac{C}{8}$ in our implementation. $\boldsymbol{p}^K$ and $\boldsymbol{p}^V$ are not shared among different attention heads. In this way, different attention heads could potentially discover various placement information, resulting in more diversified generation results. The attention output $\boldsymbol{o}$ from different heads are concatenated and transformed with another linear layer to obtain the final output $\mathbf{x}^{\text{att}} \in \mathcal{R}^{1\times C}$ of the attention layer.

Second, we apply a regression block to predict transformation parameters from the attention output. In order to generate composite images with diversified reasonable placements, we incorporate a random vector $\mathbf{z} \in \mathcal{R}^{1\times C_z}$ with dimension C_z into the block. Specifically, the regression block takes in the concatenation of $\mathbf{x}^{\text{att}}$ and $\mathbf{z}$ to predict transformation parameters $\mathbf{t} = \mathit{Regression}(\mathbf{x}^{\text{att}}, \mathbf{z})$. By sampling different $\mathbf{z}$ at test time, we can obtain a variety of reasonable $\mathbf{t}$ conditioned on $\mathbf{x}^{\text{att}}$. The detailed implementation for $\mathbf{z}$ is left to Section 3.2.

3.2 Dual-path Framework

Our whole framework is designed as a Generative Adversarial Network (GAN) [11] including a generator G and a discriminator D. The generator G is comprised of a graph completion module (GCM) and a transformation function $\mathcal{F}$. As introduced in Section 3.1, GCM works by predicting transformation parameters $\mathbf{t}$ from a tuple of $(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z})$. Using $\mathbf{t}$ as parameters for $\mathcal{F}$, we could follow Eqn.(1) to obtain a generated composite image I^{c} with object mask M^{c}. Formally, the workflow of our generator G is denoted by $(\mathrm{I}^{\mathrm{c}}, \mathrm{M}^{\mathrm{c}}) = G(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z})$. Then, the discriminator D takes the concatenated $(\mathrm{I}^{\mathrm{c}}, \mathrm{M}^{\mathrm{c}})$ as input, and predicts the probability of reasonableness for the generated composite image.

To facilitate object placement learning, we adopt a dual-path adversarial training framework containing an unsupervised path $\mathcal{P}_u$ and a supervised path $\mathcal{P}_s$. In $\mathcal{P}_u$, we only have background images I^{bg} and foreground images I^{fg} with object masks M^{fg}. In $\mathcal{P}_s$, we are provided with additional annotated composite images/masks $\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}/\mathrm{M}^{\mathrm{c}}_{\mathrm{pos}}$ (*resp.*, $\mathrm{I}^{\mathrm{c}}_{\mathrm{neg}}/\mathrm{M}^{\mathrm{c}}_{\mathrm{neg}}$) with positive (*resp.*, negative) annotation in terms of object placement, as well as their corresponding original $\mathrm{I}^{\mathrm{bg}}/\mathrm{I}^{\mathrm{fg}}/\mathrm{M}^{\mathrm{fg}}$ that constitute them. Due to the difference of provided data, $\mathcal{P}_u$ and $\mathcal{P}_s$ adopt distinct implementations for the random vector $\mathbf{z}$. To distinguish between representations in two paths, we use notation $\mathbf{z}_u$ in $\mathcal{P}_u$ and $\mathbf{z}_s$ in $\mathcal{P}_s$, respectively. The generated composite outputs in two paths are represented by $\mathrm{I}^{\mathrm{c}}_u/\mathrm{M}^{\mathrm{c}}_u$ and $\mathrm{I}^{\mathrm{c}}_s/\mathrm{M}^{\mathrm{c}}_s$ with different subscripts correspondingly.

Unsupervised Path ($\boldsymbol{\mathcal{P}_u}$). In unsupervised path $\mathcal{P}_u$, random vectors $\mathbf{z}_u$ are sampled from unit Gaussian distribution, *i.e.*, $\mathbf{z}_u \sim \mathcal{N}(\mathbf{0}, \mathbf{1})$. Correspondingly, the generator outputs $(\mathrm{I}^{\mathrm{c}}_u, \mathrm{M}^{\mathrm{c}}_u) = G(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z}_u)$. We employ an adversarial loss $\mathcal{L}^{adv}_u(G, D)$ to push the generated composite image to be undistinguishable from positive composite images.

$$\mathcal{L}^{adv}_u = \mathbb{E}_{\mathbf{z}_u \sim \mathcal{N}(\mathbf{0},\mathbf{1})}[\log(1 - D(G(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z}_u)))]. \tag{4}$$

Supervised Path ($\boldsymbol{\mathcal{P}_s}$). In supervised path $\mathcal{P}_s$, random vectors (also called latent vectors) $\mathbf{z}_s$ are designed to be sampled from global features of positive composite images via an encoder network as in VAE [15]. Given a labeled positive composite image $\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}$ with object mask $\mathrm{M}^{\mathrm{c}}_{\mathrm{pos}}$, we first calculate its ground-truth transformation parameters $\mathbf{t}_{\mathrm{gt}}$ (see details in supplementary), which obeys $(\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}, \mathrm{M}^{\mathrm{c}}_{\mathrm{pos}}) \equiv \mathcal{F}_{\mathbf{t}_{\mathrm{gt}}}(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}})$. Then, our idea is to employ the latent vector $\mathbf{z}_s$ to help reconstruct $\mathbf{t}_{\mathrm{gt}}$ because $\mathbf{z}_s$ contains potential information of positive composite images. In this way, we establish a bijection between $\mathbf{z}_s$ and $(\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}, \mathrm{M}^{\mathrm{c}}_{\mathrm{pos}})$.

Specifically, we first use the backbone network $\mathcal{E}$ to extract a feature map $\mathrm{F}^{\mathrm{c}} = \mathcal{E}(\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}, \mathrm{M}^{\mathrm{c}}_{\mathrm{pos}})$. Then we add a latent head $\mathcal{H}^{\mathrm{lat}}_{\mathbf{n}=[1]}$ to encode a global latent node $\mathbf{f}^{\mathrm{lat}} = \mathcal{H}^{\mathrm{lat}}(\mathrm{F}^{\mathrm{c}}; [1])$. After that, we employ the encoder network in VAE to sample a latent vector $\mathbf{z}_s$ from $\mathbf{f}^{\mathrm{lat}}$. Following VAE, we adopt a KL divergence loss $\mathcal{L}^{kld}_s = \mathcal{D}_{\mathrm{KL}}(\mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}_s}, \boldsymbol{\sigma}^2_{\mathbf{z}_s}) \parallel \mathcal{N}(0, 1))$ that forces the distribution of $\mathbf{z}_s$ to be

close to $\mathbf{z}_u$. With $\mathbf{z}_s$, GCM predicts transformation parameters $\mathbf{t}_s$ from a tuple of $(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z}_s)$. We utilize a reconstruction loss $\mathcal{L}_s^{rec}$ to force $\mathbf{t}_s$ to approach the ground-truth $\mathbf{t}_{\mathrm{gt}}$, which is defined as a weighted MSE between $\mathbf{t}_s$ and $\mathbf{t}_{\mathrm{gt}}$ (see details in supplementary). Then, transformation function $\mathcal{F}$ with parameters $\mathbf{t}_s$ produces $(\mathrm{I}_s^{\mathrm{c}}, \mathrm{M}_s^{\mathrm{c}})$, which is finally delivered to the discriminator D. Similar to $\mathcal{P}_u$, we also adopt an adversarial loss in $\mathcal{P}_s$:

$$\mathcal{L}_s^{adv} = \mathbb{E}_{\mathbf{z}_s \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{z}_s}, \boldsymbol{\sigma}^2_{\mathbf{z}_s})}[\log(1 - D(G(\mathrm{I}^{\mathrm{bg}}, \mathrm{I}^{\mathrm{fg}}, \mathrm{M}^{\mathrm{fg}}, \mathbf{z}_s)))]. \tag{5}$$

Additionally, we leverage both positive and negative composite images to update the discriminator by maximizing the negative form of binary cross-entropy loss:

$$\mathcal{L}_s^{cls} = \log D(\mathrm{I}^{\mathrm{c}}_{\mathrm{pos}}, \mathrm{M}^{\mathrm{c}}_{\mathrm{pos}}) + \log\left(1 - D(\mathrm{I}^{\mathrm{c}}_{\mathrm{neg}}, \mathrm{M}^{\mathrm{c}}_{\mathrm{neg}})\right). \tag{6}$$

In summary, the loss function for $\mathcal{P}_s$ is defined by

$$\mathcal{L}_s(G, D) = \mathcal{L}_s^{kld}(G) + \lambda \mathcal{L}_s^{rec}(G) + \mathcal{L}_s^{adv}(G, D) + \mathcal{L}_s^{cls}(D), \tag{7}$$

where the hyper-parameter λ is set as 50. Note that $\mathcal{L}_s^{adv}(G, D)$, $\mathcal{L}_s^{kld}(G)$, and $\mathcal{L}_s^{rec}(G)$ only handle positive composite images.

By using θ_G and θ_D to represent learnable weights in G and D, our optimization objective in the whole framework is

$$\min_{\theta_G} \max_{\theta_D} \quad \mathcal{L}_u^{adv}(G, D) + \mathcal{L}_s(G, D). \tag{8}$$

Note that two paths share weights in both G and D. Under this design, the supervised path could gradually guide the unsupervised path to generate composite images with reasonable object placement. During inference, we only use the the unsupervised path to generate composite images by sampling $\mathbf{z}_u$ from $\mathcal{N}(\mathbf{0}, \mathbf{1})$.

4 Experiment

4.1 Experimental Setting

Dataset and Evaluation Metrics. We perform experiments on OPA dataset [22], which provides binary rationality labels for composite images. The dataset includes 62074 (21376 positive / 40698 negative) composite images for training and 11396 (3588 positive / 7808 negative) composite images for testing. These annotated composite images in the dataset contain 1389 different background scenes and 4137 different foreground objects from 47 categories.

Since our objective is to generate composite images with reasonable object placements, we train our model on OPA *train* set and evaluate it on the 3588 positive samples of OPA *test* set. During inference, our model takes the foreground/background of each positive test sample as input, and generates 10 composite images by randomly sampling 10 different $\mathbf{z}_u$ in $\mathcal{P}_u$.

We adopt user study, accuracy, and FID [12] to evaluate generation plausibility and LPIPS [40] for generation diversity. User study will be introduced

Fig. 3. Visualization of object placement results. Foreground is outlined in red

in Section 4.2. We extend SimOPA [22] as a binary classifier to distinguish between reasonable and unreasonable object placements. We define accuracy as the proportion of generated composite images that are classified as positive by the binary classifier. FID is calculated between the composite images generated by our method and the positive composite images in the *test* set. We compute LPIPS for all pairs of composite images among 10 generation results for each sample, and adopt the averaged LPIPS among all samples.

Implementation Details. Our backbone network is the beginning 34 layers (including and before the fourth MaxPool layer) of VGG16 [28] with batch normalization, except that the first Conv layer has four input channels. In NEH, each node extraction layer $\mathcal{G}_l$ with parameter n_l contains three groups of (3×3 Conv, BN, ReLU), followed by a $n_l \times n_l$ AdaptiveAvgPool. The regression block contains three fully connected layers (Fc1024, Fc1024, Fc3), followed by an activation function $(\tanh(\cdot) + 1)/2$ that normalizes transformation parameters to range $(0, 1)$. We adopt the discriminator architecture in [34]. All images are resized to 256×256 and normalized before being fed into the network (see supplementary for more details). The VGG16 backbone is pretrained on ImageNet [8]. Our model is trained with batch size 32 for 11 epochs on a single RTX 3090 GPU. We adopt Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$ for optimization. The learning rate is initialized as 2×10^{-5} for backbone and discriminator, and 2×10^{-4} for the remaining parts. For hyper-parameters, we set dimension C as 512 for nodes, and C_z as 1024 for random vectors, which will be carefully analyzed in Section 4.4.

4.2 Comparison with Existing Methods

We compare our GracoNet with two baselines: TERSE [31] and PlaceNet [38]. Both of them are re-implemented on OPA dataset [22]. For the first baseline, we retain the synthesizer network and the discriminator of TERSE, and remove the target network. This is because the synthesizer is enough to generate composite images that we need, and we do not need to use the target network for downstream tasks. For the second baseline, we directly use the complete PlaceNet

Table 1. Quantitative object placement results for different methods on OPA dataset

Method	*Plausibility*			*Diversity*
	user study↑	acc.↑	FID↓	LPIPS↑
TERSE [31]	0.214	0.679	46.94	0
PlaceNet [38]	0.249	0.683	36.69	0.160
GracoNet	0.537	0.847	27.75	0.206

structure to predict object placement without further adjustment. Since TERSE and PlaceNet had been proposed before OPA dataset was released, these object placement methods did not include negative samples in their method. When we conduct experiments on OPA dataset, unless otherwise stated, we fairly use positive samples and negative samples together for both the baselines and our method. Specifically, we use negative samples in baselines by introducing a binary classification loss following Eqn.(6) to train the discriminator network.

Table 1 shows the quantitative object placement results for baselines and our proposed method. Among different evaluation metrics, user study, accuracy, and LPIPS are the most important ones. User study is conducted with 20 voluntary participants by comparing the composite images generated by TERSE, PlaceNet, and our method. For each sample, every participant chooses the method producing the most reasonable composite image. Then, each method is scored by the proportion of participants who choose it. The final score of each method is defined by the averaged score over all samples.

By comparison, our method significantly outperforms TERSE/PlaceNet in generation plausibility (0.537 *v.s.* 0.214/0.249 for user study, 0.847 *v.s.* 0.679/0.683 for accuracy, and 27.75 *v.s.* 46.94/36.69 for FID). Also, our method achieves better LPIPS in generation diversity than PlaceNet. Note that TERSE does not incorporate randomness, so its generation diversity is zero. Generally, our method performs satisfactorily and balances plausibility and diversity well.

Figure 3 visualizes some object placement results for different methods. As illustrated, our method works better in predicting foreground locations and sizes by comprehensively analyzing different background regions, verifying the effectiveness of our designed GCM. In the supplementary, we show more visualizations of generation plausibility and diversity from three aspects: 1) combination of an identical background scene and different foreground objects, 2) combination of an identical foreground object and different background scenes, and 3) sampling different random vectors for the same pair of foreground and background.

4.3 Ablation Studies

Different Choices of Background Head $\mathcal{H}^{\mathbf{bg}}$. Table 2 displays four choices of parameter $\mathbf{n}$ in the background head $\mathcal{H}^{\mathrm{bg}}$. Since the input size of images is 256×256 and the backbone $\mathcal{E}$ contains four pooling layers, feature maps F are 16×16 in size. As default, we choose $\mathbf{n} = [2, 4, 8]$ to represent large-scale, medium-scale, and small-scale background nodes, which pay attention to different regions

Table 2. Ablation study on background head $\mathcal{H}^{\text{bg}}$

$\mathcal{H}^{\text{bg}}$ with parameter $\mathbf{n}$	*Plausibility*		*Diversity*
	acc.↑	FID↓	LPIPS↑
$\mathbf{n} = [2]$	0.807	37.44	0.136
$\mathbf{n} = [2, 4]$	0.821	34.23	0.146
$\mathbf{n} = [2, 4, 8, 16]$	0.837	25.91	0.154
$\mathbf{n} = [2, 4, 8]$	0.847	27.75	0.206

Table 3. Ablation study on different types of learnable placement encodings

$\boldsymbol{p}^K$	$\boldsymbol{p}^V$	shared across heads	*Plausibility*		*Diversity*
			acc.↑	FID↓	LPIPS↑
		-	0.793	28.22	0.120
✓		✓	0.809	28.02	0.133
	✓	✓	0.844	25.14	0.127
✓	✓	✓	0.851	29.09	0.170
✓			0.836	26.85	0.156
	✓		0.839	26.22	0.154
✓	✓		0.847	27.75	0.206

on the feature map. In detail, each of the 4 large-scale (*resp.*, 16 medium-scale, 64 small-scale) background nodes focuses on a local receptive field of 8×8 (*resp.*, 4×4, 2×2) region on F. In Table 2, we show more choices of parameter $\mathbf{n}$ in $\mathcal{H}^{\text{bg}}$. When $\mathbf{n} = [2]$ or $\mathbf{n} = [2, 4]$, the model misses small-scale information, so the placement process lacks details in some local regions. When $\mathbf{n} = [2, 4, 8, 16]$, the model tends to learn pixel-wise knowledge, which is redundant in object placement learning and adversely affects network optimization. Overall, our choice $\mathbf{n} = [2, 4, 8]$ achieves a good balance.

Different Types of Learnable Placement Encoding. As discussed in Section 3.1, we adopt placement encoding $\boldsymbol{p}^K$ and $\boldsymbol{p}^V$ in the attention layer. In our implementation, $\boldsymbol{p}^K$ and $\boldsymbol{p}^V$ are both learnable and not shared across different attention heads. In this paragraph, we discuss more variants of placement encoding, as shown in Table 3. Without placement encoding, the generation plausibility and diversity witness a considerable decrease, because location and size information of different background nodes are missing under this condition. With a single $\boldsymbol{p}^K$ or $\boldsymbol{p}^V$, the model performs a little better in plausibility, but the diversity is still unsatisfactory. Using $\boldsymbol{p}^K$ and $\boldsymbol{p}^V$ together works best. If placement encodings are shared across attention heads, different attention heads could not learn diversified attention regions for object placement, resulting in comparably low generation diversity. Therefore, we choose to use $\boldsymbol{p}^K$ and $\boldsymbol{p}^V$ together, and make them independent across different attention heads. Comprehensively, our choice considers both plausibility and diversity to achieve the best results.

Table 4. Ablation study on functionality of GCM

Fg/Bg Feature Extractor	Placement Finder	*Plausibility* acc.↑	*Plausibility* FID↓	*Diversity* LPIPS↑
global vector	concat+fc	0.793	33.97	0.082
NEH	concat+fc	0.828	31.36	0.144
NEH	PSN	0.847	27.75	0.206

Table 5. Ablation study on loss functions

Method	*Plausibility* acc.↑	*Plausibility* FID↓	*Diversity* LPIPS↑
w/o $\mathcal{L}_u^{adv}$	0.790	32.64	0.038
w/o $\mathcal{L}_s^{adv}$	0.800	34.76	0.057
w/o $\mathcal{L}_s^{cls}$	0.734	34.62	0.033
w/o $\mathcal{L}_s^{kld}$	0.844	29.26	0.199
w/o $\mathcal{L}_s^{rec}$	0.767	25.53	0.131
Ours	0.847	27.75	0.206

Functionality of GCM. As introduced in Section 3.1, our graph completion module (GCM) consists of two important components: node extraction head (NEH) and placement seeking network (PSN). Table 4 provides an ablation study on GCM by replacing NEH or PSN with naive network structures. In Table 4, the first experiment uses split branches to extract global features for foreground and background respectively. Then the two global features are concatenated and regressed with several fc layers to obtain the transformation parameters for object placement. Compared with the first experiment, the second one uses NEH to extract features by considering different background locations and sizes. The background nodes are then averaged and concatenated with the foreground node to predict transformation parameters. The last experiment is our method that combines NEH for feature extraction and PSN for placement finding. Comparing the results of the first two experiments, we find that using NEH to explicitly encode different background locations and sizes is beneficial for object placement. By comparing the results of the last two experiments, PSN works better in discovering relationships between foreground and background, and successfully establishes a reasonable connection between foreground node and background nodes in the node graph. In summary, both NEH and PSN are crucial in our method. They work together to ensure the functionality of our proposed GCM.

Utility of Different Loss Functions. Table 5 gives an ablation study on different loss functions. Except for $\mathcal{L}_s^{kld}$, deleting any loss makes the performance drop sharply on plausibility or diversity, especially for $\mathcal{L}_s^{cls}$ and $\mathcal{L}_s^{rec}$. Although our model still performs passably without $\mathcal{L}_s^{kld}$, adding this loss can bring some improvement. Generally, every loss makes up an important part, and they work together to guarantee the effectiveness of our method.

Table 6. Choices of hyper-parameter C_z

C_z	*Plausibility*		*Diversity*
	acc.↑	FID↓	LPIPS↑
256	0.807	36.04	0.151
512	0.838	35.53	0.175
2048	0.827	28.10	0.208
4096	0.792	27.39	0.219
1024	0.847	27.75	0.206

Table 7. Choices of hyper-parameter λ

λ	*Plausibility*		*Diversity*
	acc.↑	FID↓	LPIPS↑
1	0.820	23.04	0.183
10	0.823	29.93	0.190
25	0.847	28.57	0.193
100	0.821	24.62	0.174
50	0.847	27.75	0.206

4.4 Hyper-parameter Analyses

Dimension C_z of Random and Latent Vectors. As introduced in Section 4.1, we set the dimension of nodes as $C = 512$, and set the dimension of random vector $\mathbf{z}_u$ and latent vector $\mathbf{z}_s$ as $C_z = 1024$. The value of C is chosen by considering both hardware resource occupation and model performance. Table 6 analyzes different choices of C_z in the range of $[256, 4096]$. When C_z increases, accuracy first increases, meets a peak at 1024, and then decreases. Meanwhile, LPIPS increases and FID decreases generally. For a balanced consideration between plausibility and diversity, we choose $C_z = 1024$ in our implementation.

Coefficient λ of Reconstruction Loss. As discussed in Section 4.3, reconstruction loss $\mathcal{L}_s^{rec}$ plays an important role in the supervised path. It helps our model reconstruct ground-truth transformation parameters from positive composite images, enabling the supervised path to guide the unsupervised path through the unified training of two paths. In Eqn.(7), reconstruction loss $\mathcal{L}_s^{rec}$ has a hyper-parameter λ indicating the coefficient/weight of this loss during optimization. Table 7 displays different choices of λ in the range of $[1, 100]$. As can be analyzed, our model achieves comparably good plausibility and diversity when $\lambda = 50$, which becomes our default choice in all experiments.

4.5 Visualization of Multi-head Attention

In Section 3.1, we introduce multi-head attention with placement encoding to discover the relationship between the unique foreground node and totally 84 background nodes. Eqn.(3) defines attention coefficient α_j $(1 \leqslant j \leqslant 84)$ in each head to represent the edge weight between the foreground node and the j-th background node in the node graph. Specifically, we use $j_{\max}$ to denote the index with the maximum edge weight:

$$j_{\max} = \arg\max_j \alpha_j \qquad j = 1, 2, \cdots, 84. \tag{9}$$

Since we have 8 attention heads, we use $j_{\max}^{(k)}$ with $1 \leqslant k \leqslant 8$ for differentiated representations in distinct heads. In the k-th attention head, $j_{\max}^{(k)}$ corresponds to a local background region with a specific location and size encoded by the $j_{\max}$-th

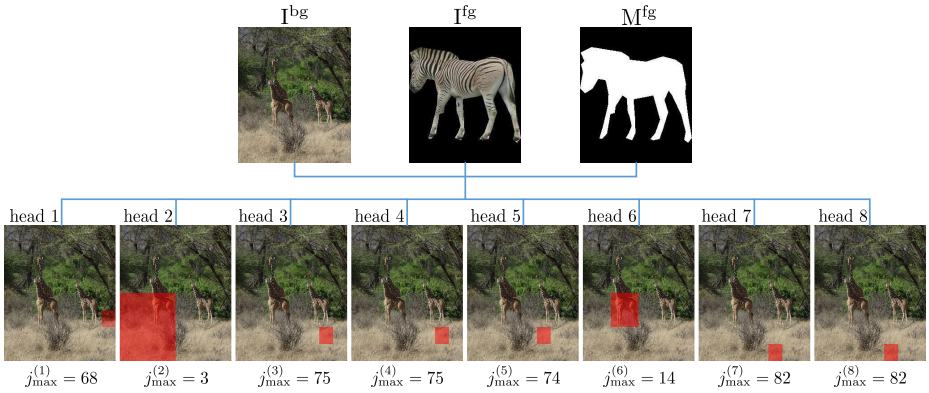

Fig. 4. Visualization of background regions (marked in red) that the multi-head attention layer pays the most attention to in each head. $j^{(k)}_{\max}$ is the index of background node with the maximum attention coefficient in the k-th head. See details in Section 4.5

background node. According to the definition of background nodes in Section 3.1, $j^{(k)}_{\max}$ corresponds to a local background region with scale $\frac{H}{2} \times \frac{H}{2}$ (*resp.*, $\frac{H}{4} \times \frac{H}{4}$, $\frac{H}{8} \times \frac{H}{8}$) when $1 \leqslant j^{(k)}_{\max} \leqslant 4$ (*resp.*, $5 \leqslant j^{(k)}_{\max} \leqslant 20,\ 21 \leqslant j^{(k)}_{\max} \leqslant 84$). In Figure 4, we visualize the local background region attended by $j^{(k)}_{\max}$ for each attention head, which represents for the region that the k-th head pays the most attention to on the background scene. As illustrated, the multi-head attention layer successfully discovers diversified locations and sizes of potential background regions for object placement by learning the relationship between the unique foreground node and multiple background nodes. We could also conclude from Figure 4 that the most attended regions are reasonable enough for placing the zebra because only near-earth locations are activated, which accords with common sense.

5 Conclusion

In this work, we have proposed a novel graph completion module (GCM) for the object placement task to explicitly explore the relationship between the foreground object and the background image. We have also designed a dual-path framework upon the GCM structure, in which the supervised path provides additional cues for the unsupervised path so as to significantly enhance the performance. Extensive experiments on OPA dataset have demonstrated the effectiveness of our proposed method.

Acknowledgements The work is supported by Shanghai Municipal Science and Technology Key Project (Grant No. 20511100300), Shanghai Municipal Science and Technology Major Project, China (2021SHZDZX0102), and National Science Foundation of China (Grant No. 61902247).

References

1. Azadi, S., Pathak, D., Ebrahimi, S., Darrell, T.: Compositional gan: Learning image-conditional binary composition. International Journal of Computer Vision **128**, 2570–2585 (2020)
2. Chen, B.C., Kae, A.: Toward realistic image compositing with adversarial learning. In: CVPR (2019)
3. Chen, T., Cheng, M.M., Tan, P., Shamir, A., Hu, S.M.: Sketch2photo: Internet image montage. ACM transactions on graphics (TOG) **28**, 1–10 (2009)
4. Cong, W., Niu, L., Zhang, J., Liang, J., Zhang, L.: Bargainnet: Background-guided domain translation for image harmonization. In: ICME (2021)
5. Cong, W., Tao, X., Niu, L., Liang, J., Gao, X., Sun, Q., Zhang, L.: High-resolution image harmonization via collaborative dual transformations. In: CVPR (2022)
6. Cong, W., Zhang, J., Niu, L., Liu, L., Ling, Z., Li, W., Zhang, L.: Dovenet: Deep image harmonization via domain verification. In: CVPR (2020)
7. Dai, Z., Yang, Z., Yang, Y., Carbonell, J., Le, Q.V., Salakhutdinov, R.: Transformer-xl: Attentive language models beyond a fixed-length context (2019)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: CVPR (2009)
9. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: ICML (2017)
10. Georgakis, G., Mousavian, A., Berg, A.C., Kosecka, J.: Synthesizing training data for object detection in indoor scenes (2017)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. NIPS (2014)
12. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. NeurIPS (2017)
13. Hong, Y., Niu, L., Zhang, J.: Shadow generation for composite image in real-world scenes. In: AAAI (2022)
14. Johnson, J., Gupta, A., Fei-Fei, L.: Image generation from scene graphs. In: CVPR (2018)
15. Kingma, D.P., Welling, M.: Auto-encoding variational bayes (2014)
16. Lalonde, J.F., Efros, A.A.: Using color compatibility for assessing image realism. In: ICCV (2007)
17. Lalonde, J.F., Hoiem, D., Efros, A.A., Rother, C., Winn, J., Criminisi, A.: Photo clip art. ACM transactions on graphics (TOG) **26**, 3–es (2007)
18. Lee, D., Liu, S., Gu, J., Liu, M.Y., Yang, M.H., Kautz, J.: Context-aware synthesis and placement of object instances (2018)
19. Li, X., Liu, S., Kim, K., Wang, X., Yang, M.H., Kautz, J.: Putting humans in a scene: Learning affordance in 3d indoor environments. In: CVPR (2019)
20. Lin, C.H., Yumer, E., Wang, O., Shechtman, E., Lucey, S.: St-gan: Spatial transformer generative adversarial networks for image compositing. In: CVPR (2018)
21. Liu, D., Long, C., Zhang, H., Yu, H., Dong, X., Xiao, C.: Arshadowgan: Shadow generative adversarial network for augmented reality in single light scenes. In: CVPR (2020)
22. Liu, L., Zhang, B., Li, J., Niu, L., Liu, Q., Zhang, L.: OPA: Object placement assessment dataset. arXiv preprint arXiv:2107.01889 (2021)
23. Liu, X., Yu, H.F., Dhillon, I., Hsieh, C.J.: Learning to encode position for transformer with continuous dynamical model. In: ICML (2020)

24. Niu, L., Cong, W., Liu, L., Hong, Y., Zhang, B., Liang, J., Zhang, L.: Making images real again: A comprehensive survey on deep image composition. arXiv preprint arXiv:2106.14490 (2021)
25. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer (2020)
26. Schuster, M.J., Okerman, J., Nguyen, H., Rehg, J.M., Kemp, C.C.: Perceiving clutter and surfaces for object placement in indoor environments. In: ICHR (2010)
27. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations (2018)
28. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2015)
29. Smith, A.R., Blinn, J.F.: Blue screen matting. In: SIGGRAPH (1996)
30. Tan, F., Bernier, C., Cohen, B., Ordonez, V., Barnes, C.: Where and who? automatic semantic-aware person composition. In: WACV (2018)
31. Tripathi, S., Chandra, S., Agrawal, A., Tyagi, A., Rehg, J.M., Chari, V.: Learning to generate synthetic data via compositing. In: CVPR (2019)
32. Tsai, Y.H., Shen, X., Lin, Z., Sunkavalli, K., Lu, X., Yang, M.H.: Deep image harmonization. In: CVPR (2017)
33. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS (2017)
34. Wang, T.C., Liu, M.Y., Zhu, J.Y., Tao, A., Kautz, J., Catanzaro, B.: High-resolution image synthesis and semantic manipulation with conditional gans. In: CVPR (2018)
35. Weng, S., Li, W., Li, D., Jin, H., Shi, B.: Misc: Multi-condition injection and spatially-adaptive compositing for conditional person image synthesis. In: CVPR (2020)
36. Wu, H., Zheng, S., Zhang, J., Huang, K.: Gp-gan: Towards realistic high-resolution image blending. In: ACM Multimedia (2019)
37. Xue, S., Agarwala, A., Dorsey, J., Rushmeier, H.: Understanding and improving the realism of image composites. ACM Transactions on graphics (TOG) **31**, 1–10 (2012)
38. Zhang, L., Wen, T., Min, J., Wang, J., Han, D., Shi, J.: Learning object placement by inpainting for compositional data augmentation. In: ECCV (2020)
39. Zhang, L., Wen, T., Shi, J.: Deep image blending. In: WACV (2020)
40. Zhang, R., Isola, P., Efros, A.A., Shechtman, E., Wang, O.: The unreasonable effectiveness of deep features as a perceptual metric. In: CVPR (2018)
41. Zhang, S.H., Zhou, Z.P., Liu, B., Dong, X., Hall, P.: What and where: A context-based recommendation system for object insertion. Computational Visual Media **6**, 79–93 (2020)
42. Zhu, J.Y., Krahenbuhl, P., Shechtman, E., Efros, A.A.: Learning a discriminative model for the perception of realism in composite images. In: ICCV. pp. 3943–3951 (2015)
43. Zhu, J.Y., Zhang, R., Pathak, D., Darrell, T., Efros, A.A., Wang, O., Shechtman, E.: Multimodal image-to-image translation by enforcing bi-cycle consistency. In: NeurIPS (2017)