Image Generation from a Hyper Scene Graph with Trinomial Hyperedges

Ryosuke Miyake, Tetsu Matsukawa[a] and Einoshin Suzuki[b]

Graduate School and Faculty of Information Science and Electrical Engineering, Kyushu University, Fukuoka, Japan

Keywords: Image Generation, Scene Graph, Hyper Graph, Generative Adversarial Network.

Abstract: Generating realistic images is one of the important problems in the field of computer vision. In image generation tasks, generating images consistent with an input given by the user is called conditional image generation. Due to the recent advances in generating high-quality images with Generative Adversarial Networks, many conditional image generation models have been proposed, such as text-to-image, scene-graph-to-image, and layout-to-image models. Among them, scene-graph-to-image models have the advantage of generating an image for a complex situation according to the structure of a scene graph. However, existing scene-graph-to-image models have difficulty in capturing positional relations among three or more objects since a scene graph can only represent relations between two objects. In this paper, we propose a novel image generation model which addresses this shortcoming by generating images from a hyper scene graph with trinomial edges. We also use a layout-to-image model supplementally to generate higher resolution images. Experimental validations on COCO-Stuff and Visual Genome datasets show that the proposed model generates more natural and faithful images to user's inputs than a cutting-edge scene-graph-to-image model.

1 INTRODUCTION

Generating realistic images is one of the important problems in the field of computer vision. Image generation can be applied in various fields (Agnese et al., 2020; Wu et al., 2017) such as medicine (Nie et al., 2017; Ghorbani et al., 2020) and art (Elgammal et al., 2017). In the field of art, it could be useful for artists and graphic designers. In the future, when higher-quality images can be generated, image or video search engines can be replaced by algorithms which generate customized content based on user preferences (Johnson et al., 2018).

In image generation tasks, generating images consistent with the input given by the user is called conditional image generation. In recent years, advances in research on Generative Adversarial Networks (GAN), (Goodfellow et al., 2014) have improved the quality of generated images, and many conditional image generation models have been proposed. Among them, text-to-image models (Reed et al., 2016; Zhang et al., 2017; Zhang et al., 2018; Odena et al., 2017) generate images conditioned on a text such as "the sheep is on the grass". These models have the advantage of their simple input and their ease of application in many fields. Meanwhile, they have the disadvantage of their difficulty in generating images which represent complex situations with many objects and their relations. This drawback is attributed to the difficulty in generating a single feature vector which contains all of the information in a long sentence.

A scene graph represents situations in a similar way to a text (Johnson et al., 2015). It consists of nodes representing objects such as "dog" and binomial edges representing a relation between two objects such as "on". In the scene-graph-to-image models, each object or edge label in a scene graph is transformed into a feature vector. In other words, text-to-image models convert a sentence into a feature vector, while scene-graph-to-image models generate feature vectors from each word in a scene graph. Thus, encoding a scene graph into a feature vector is easier, and this fact enables these models to generate proper images for complex situations.

Though scene-graph-to-image models have such an advantage, they have a shortcoming: the positional relations among three or more objects tend to be inaccurate (Figure1 (a)). An edge in a scene graph can represent only relations between two objects, so it has difficulty in capturing a positional relation among three or more objects. This inaccurate object position-

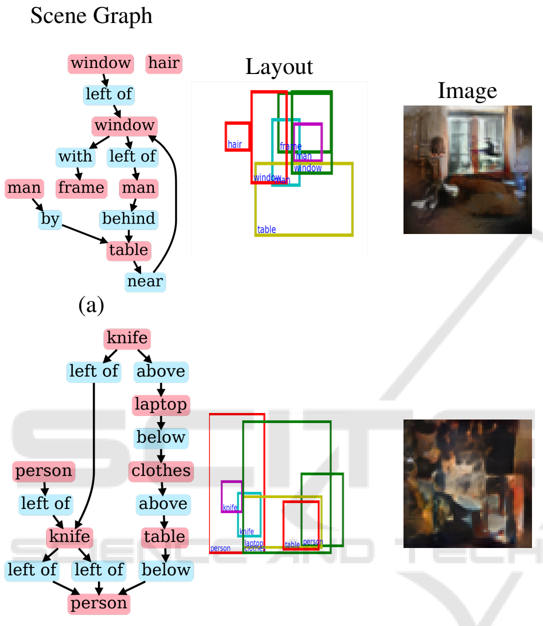
[a] https://orcid.org/0000-0002-8841-6304

[b] https://orcid.org/0000-0001-7743-6177

Miyake, R., Matsukawa, T. and Suzuki, E.
Image Generation from a Hyper Scene Graph with Trinomial Hyperedges.
DOI: 10.5220/0011699300003417
In *Proceedings of the 18th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (VISIGRAPP 2023) - Volume 5: VISAPP*, pages 185-195
ISBN: 978-989-758-634-7; ISSN: 2184-4321

ing also causes overlapping of positions of the generated objects, which causes another shortcoming of generating unclear objects (Figure 1 (b)).

We address the former shortcoming by generating images from the hyper scene graph with trinomial hyperedges each of which represents the positional relation among three objects. If the shortcoming of the positional relationship is improved with hyperedges, the object overlapping is reduced, resulting in sharper objects and images. Moreover, we use layout-to-image model Layout2img (He et al., 2021) supplementally to generate high resolution images.



(a)

(b)

Figure 1: Example of scene-graph-to-model sg2im (Johnson et al., 2018) failing to generate images which are not consistent with the input. The layout in the second column is the intermediate product of sg2im. The scene graph in (a) has a path "window" (green bounding box) → "left of" → "man" (purple). However, in the generated layout, "window" (green) surround "man" (purple) and is not positioned in the left of "man" (purple). In the generated image in (b), the entire image is unclear, and thus it is difficult to insist that the objects in the scene graph are generated.

2 RELATED WORKS

Image generation models fall into three categories: (i) GAN (Goodfellow et al., 2014; Radford et al., 2015), (ii) VAE (Kingma and Welling, 2013), and (iii) autoregressive models (Van Den Oord et al., 2016). (i) A GAN consists of a generator G and a discriminator D. The generator generates data x' from noise z sampled from noise distribution p_z. The discriminator outputs the probability that the given data is not x' but x sampled from the training data. These two models are trained to compete with each other. (ii) VAE consists of an encoder and a decoder. The encoder takes an image as input and outputs a latent variable. The decoder aims to recover the original image from the latent variable. The two networks are trained simultaneously. (iii) The autoregressive model generates an image by sequentially generating the value of each pixel conditioned on all previously generated pixels.

In these three models, GAN is widely used in conditional image generation models due to the realistic look of the generated images and its ease of application to various models. In this paper, we also employ GAN as the generative model.

Major inputs for conditional image generation models using GAN are (1) text (Reed et al., 2016; Zhang et al., 2017; Zhang et al., 2018; Odena et al., 2017), (2) layouts (Sun and Wu, 2019; He et al., 2021; Hinz et al., 2019; Hinz et al., 2022), and (3) scene graphs (Johnson et al., 2018; Li et al., 2019; Vo and Sugimoto, 2020; Mittal et al., 2019). In the following, we categorize each generative model by these three kinds of inputs and explain them.

(1) The text-to-image models have been studied extensively due to their advantages of their simple input and their ease of application in many fields. Reed et al. extended cGAN (Mirza and Osindero, 2014) to generate images which are aligned with the input semantically by using GAN-INT-CLS (Reed et al., 2016). Odena et al. proposed AC-GAN (Odena et al., 2017), in which the discriminator solves a task of identifying the class of a given image in addition to the general discriminative problem, and tried to generate images where each object can be identified. These models can generate an image consistent with the input for a simple situation which involves few objects and few relations. However, they have difficulty in generating images representing complex situations with many objects and relations between them.

(2) Layout-to-image models and (3) scene-graph-to-image models were proposed to overcome the shortcoming of text-to-image models (Johnson et al., 2018; Hinz et al., 2019). (2) He et al. proposed a layout-to-image model Layout2img (He et al., 2021). This model generates consistent feature vectors for each object and generates natural images. The layout-to-image models can control the position of the generated object directly by the input. On the other hand, they have difficulty in being applied to image generation from a text due to the dissimilarity of the structures between a text and a layout.

(3) Johnson et al. proposed a scene-graph-to-image model sg2im (Johnson et al., 2018). This

model processes a scene graph using a graph convolutional network. Compared with the text-to-image model StackGAN (Zhang et al., 2017), sg2im generates images which are semantically more consistent with the input. Scene-graph-to-image models possess two main advantages, i.e., the ease of converting the input to a feature vector and the ease of applying them to text-to-image models. The former is due to their ease of making feature vectors. A text-to-image model converts an entire sentence into a feature vector, while a scene-graph-to-image model converts each word into a feature vector. The latter is attributed to the similarity of the structures between a text and a scene graph. Actually, there is research on the transformation from a text to a scene graph (Schuster et al., 2015). On the other hand, as explained in Section 1, the scene-graph-to-image model has difficulty in capturing the positional relations among three or more objects correctly.

In this paper, we focus on scene-graph-to-image models due to their advantages and aim to overcome its shortcoming. In addition, we use the layout-to-image model as a supplement to generate higher resolution images.

3 TARGET PROBLEM

To make the distinction between Johnson et al.'s work and ours clear, first, we explain their target problem (Johnson et al., 2018): image generation from a scene graph. Then, we describe our target problem: image generation from a hyper scene graph, and the evaluation metrics used in this paper.

3.1 Image Generation from a Scene Graph

Johnson et al. set the target problem to creating generator $G(S,z)$ which generates image $\hat{I}$ from scene graph $S=(V,E)$ and Gaussian noise z. Each object $v_i \in V$ has a category such as "dog" and "sky". V denotes the set of nodes in the scene graph, where $V=\{v_1,\ldots,v_n\}$, and n represents the number of nodes. A node represents an object. Category $c_i \in \mathcal{C}$ of object v_i is denoted by $c_i = f(v_i)$, where $\mathcal{C}$ is the set of all categories of objects and $f(\cdot)$ is a mapping from object $v \in V$ to category $c \in \mathcal{C}$ of objects. E denotes the set of binomial edges in a scene graph, satisfying $E \subseteq V \times \mathcal{R}_2 \times V$, where $\mathcal{R}_2$ is the entire set of labels for binomial relations ("on", "left of", etc.). Note that for $(v_i, r_j, v_k) \in E$, $i \neq k$. A binomial edge is directed: (v_i, r_j, v_k) and (v_k, r_j, v_i) are distinct. Figure 2 (a) shows an example of a scene graph.

3.2 Image Generation from a Hyper Scene Graph

We define the hyper scene graph as a scene graph with an additional hyperedge that represents a relation among three or more objects. In this paper, we focus on relations among three objects for simplicity. As described in Section 1, sg2im (Johnson et al., 2018) has a shortcoming: the positional relations in the generated image among three or more objects tend to be inaccurate. To address this shortcoming, we set our target problem to creating generator $G(H,z)$ which generates image $\hat{I}$ from hyper scene graph $H=(V,E,Q)$ and Gaussian noise z. Q denotes the set of trinomial hyperdeges in H, which satisfies $Q \subseteq V \times \mathcal{R}_3 \times V \times V$. $\mathcal{R}_3$ denotes the entire set of labels (such as "between") for the trinomial relation. Trinomial hyperedge $(v_i, r_j, v_k, v_l) \in Q$ satisfies $i \neq k$, $i \neq l$, $k \neq l$ and is directed: (v_i, r_j, v_k, v_l), (v_i, r_j, v_l, v_k), (v_l, r_j, v_i, v_k), (v_l, r_j, v_k, v_i), (v_k, r_j, v_i, v_l), and (v_k, r_j, v_l, v_i) are distinct. The definitions of V and E are the same as in Section 3.1. Figure 2 (b) shows an example of a hyper scene graph.

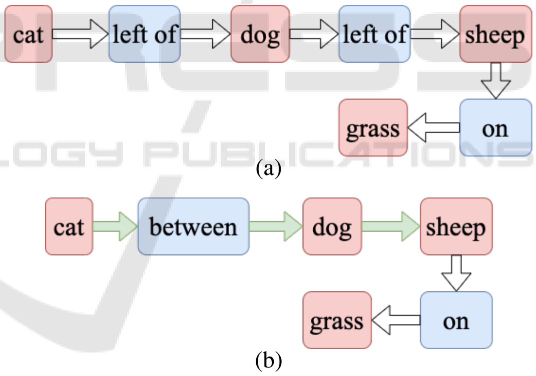


Figure 2: Example of a scene graph (a) and a hyper scene graph (b). The red box and blue box represent an object and an edge label, respectively. The white arrow and green arrow represent a binomial edge and a trinomial hyperedge, respectively. Set V of objects and category $f(V)$ of objects are the same in both (a) and (b), and they are given by $V=\{v_1,v_2,v_3,v_4\}$, $f(V)=\{\text{“cat”},\text{“dog”},\text{“sheep”},\text{“grass”}\}$. Set E_{a} and E_{b} of binomial edges are given by $E_{\mathrm{a}}=\{(v_1,r_1(\text{“left of”}),\mathrm{v}_2)$, $(v_2,r_2(\text{“left of”}),v_3)$, $(v_3,r_3(\text{“on”}),v_4)\}$ and $E_{\mathrm{b}}=\{(v_3,r_3(\text{“on”}),v_4)\}$, respectively. Set Q_{b} of trinomial hypereges is given by $Q_{\mathrm{b}}=\{(v_1,r_4(\text{“between”}),v_2,v_3)\}$. The path "cat" → "left of" → "dog" corresponds to the binomial edge $(v_1(\text{“cat”})),\text{“left of”},v_2(\text{“dog”}))$ and means that a cat exists to the left of a dog. Also, the path "cat" → "between " → "dog" → "sheep" corresponds to the trinomial edge $(v_1(\text{“cat”})),\text{“between”},v_2(\text{“dog”}),v_2(\text{“sheep”}))$ and means from left to right, a cat, a dog, and a sheep align in a row.

3.3 Evaluation Metrics

There are various evaluation measures for conditional image generation models, such as the fidelity and the diversity of the generated images to the input, the clarity of the boundaries, and the robustness to small changes in the input (Frolov et al., 2021). In this paper, we aim to achieve the following three goals.

A). Improving the positional relation among three objects connected to a hyperedge.

B). Making the overlapping of the objects small.

C). Making the generated image natural.

As the evaluation metrics for (A)∼(C), we use Positional relation of Three Objects (PTO), Area of Overlapping (AoO), and Inception Score (Salimans et al., 2016), respectively. PTO and AoO are new measures proposed in this paper. As an evaluation measure for (C), we could also use the Fréchet Inception Distance (FID) (Heusel et al., 2017), which is the distance between the distribution of the embedded representations of the real images and the generated images. However, we use Inception Score in line with (Johnson et al., 2018) in this paper.

PTO is the percentage of correctly generating three bounding boxes b_i, b_j, b_k for v_i, v_j, v_k satisfying $(v_i, \text{“between”}, v_j, v_k) \in Q$. We judge that bounding boxes b_i, b_j, b_k are correctly generated if they satisfy the following five conditions for the addition of a trinomial hyperedge “between”.

- v_i, v_j, v_k are lined up from left to right in this order, where each of objects does not overlapping, i.e., $x_{i0} < x_{i1} < x_{j0} < x_{j1} < x_{k0} < x_{k1}$.
- v_i, v_j, v_k are not large objects such as the background, i.e., $x_{i1} - x_{i0} < 0.7w$ and $x_{k1} - x_{k0} < 0.7w$ and $x_{k1} - x_{k0} < 0.7w$, where w is the image width.
- v_i, v_j, v_k are of nearly equal size, i.e., $\frac{1}{2} < \frac{x_{i1}-x_{i0}}{x_{j1}-x_{j0}} < 2$ and $\frac{1}{2} < \frac{y_{i1}-y_{i0}}{y_{j1}-y_{j0}} < 2$ and $\frac{1}{2} < \frac{x_{j1}-x_{j0}}{x_{k1}-x_{k0}} < 2$ and $\frac{1}{2} < \frac{y_{j1}-y_{j0}}{y_{k1}-y_{k0}} < 2$.
- v_i, v_j, v_k are not largely apart horizontally, i.e., $\max(x_{i1} - x_{i0}, x_{j1} - x_{j0})0.5 > x_{j0} - x_{i1}$ and $\max(x_{j1} - x_{j0}, x_{k1} - x_{k0})0.5 > x_{k0} - x_{j1}$.
- v_i, v_j, v_k are not largely apart vertically, i.e., $\max(y_{i1} - y_{i0}, y_{j1} - y_{j0})0.7 > y_{j0} - y_{i1}$ and $\max(y_{j1} - y_{j0}, y_{k1} - y_{k0})0.7 > y_{k0} - y_{j1}$.

Here, $b_i = (x_{i0}, x_{i1}, y_{i0}, y_{i1})$, $(x_{i0} < x_{i1}, y_{i0} < y_{i1})$, which means that the bounding box is a rectangle with vertices $(x_{i0}, y_{i0}), (x_{i0}, y_{i1}), (x_{i1}, y_{i0}), (x_{i1}, y_{i1})$. PTO is defined by

$$\text{PTO} = \frac{\text{the number of correctly generated sets}}{\text{the number of the evaluated sets}}. \quad (1)$$

For sg2im (Johnson et al., 2018), which employs no hyperedges, we evaluate whether objects (v_i, v_j, v_k) satisfy $(v_i, \text{“left of”}, v_j)$ and $(v_i, \text{“left of”}, v_j) \in E$.

AoO examines whether the introduction of the trinomial hyperedge reduces the overlapping of objects. AoU measures the overlapping of three objects connected to a trinomial hyperedge. AoU is defined as follows.

$$\text{AoO}(b_i, b_j, b_k) = \text{IoU}(b_i, b_j) + \text{IoU}(b_i, b_k) + \text{IoU}(b_j, b_k) \quad (2)$$

where IoU (Intersection over Union) (Rezatofighi et al., 2019) is a measure for evaluating the overlapping of two bounding boxes. IoU has been used to evaluate the overlapping of generated boxes and the ground truth boxes in the dataset. However, our objective is to improve the relative position between the generated bounding boxes as we explained in Section 1. Thus, we use IoU for the evaluation of the overlapping of three objects here. Let the two bounding boxes be X, Y and $S(\cdot)$ be the area of the input region. Then, IoU is defined as follows.

$$\text{IoU}(X, Y) = \frac{S(X \cap Y)}{S(X \cup Y)} \quad (3)$$

The smaller AoO is, the smaller the overlapping between the objects is, which indicates that the objects are generated with appropriate positioning. In sg2im, we evaluate whether the bounding boxes of v_i, v_j, v_k satisfy $(v_i, \text{“left of”}, v_j)$, $(v_i, \text{“left of”}, v_j) \in E$.

We use Inception Score (IS) (Salimans et al., 2016) as a measure for evaluating the naturalness of the generated image. IS is calculated using Inception Network trained on ImageNet (Russakovsky et al., 2015; Szegedy et al., 2015) with the following equations.

$$\text{IS} = \exp[\mathrm{E}_{\hat{I}}[\mathrm{D_{KL}}(p(y|\hat{I}) \| p(y))]], \quad (4)$$

$$p(y) = \frac{1}{N} \sum_{i=1}^{N} p(y|\hat{I}_i), \quad (5)$$

where $p(y|\hat{I})$ is the probability of label y of input image $\hat{I}$ predicted by Inception Network and $p(y)$ is its marginal probability. The score becomes larger when the labels of the generated images are easily identified and the identified labels are diverse. Therefore, we can use Inception Score as the evaluation measure for the naturalness of the generated image.

4 ORIGINAL MODEL

We design our model based on an image generation model from a scene graph sg2im (Johnson et al., 2018). In this Section, we explain sg2im (Johnson et al., 2018) and its shortcoming.

4.1 sg2im (Johnson et al., 2018)

First, we show an overview of generator G of sg2im (Johnson et al., 2018) in Figure 3 (a). In the generator of sg2im, a hyper scene graph is a scene graph, the hyper graph convolutional network is the graph convolutional network, a layout is a feature map, and Layout2img is Cascaded Refinement Network (Chen and Koltun, 2017). The image generation in the generation phase is performed as follows.

1. Each $v \in V$ of the objects and binomial relation labels r $((\cdot, r, \cdot) \in E)$ in scene graph S is transformed into object vector ν and relation vector ρ by the object embedding network and the relation embedding network, respectively.
2. We apply graph convolution on object vector ν and relation vector ρ based on scene graph S and obtain convoluted object vectors.
3. The box regression network is applied to the convoluted object vectors to predict bounding box $\hat{B}$ which represents the region to generate each object.
4. We generate feature map M, which is an intermediate representation between a scene graph and an image region, by mapping the convoluted object vectors based on their bounding boxes $\hat{B}$.
5. Feature map M with Gaussian noise z is fed into the Cascaded Refinement Network (Chen and Koltun, 2017) to generate image $\hat{I}$.

The graph convolution in step 2 is performed to convert the feature vector of each object so that it considers the entire scene graph. Figure 4 shows the process flow of the first layer of the graph convolution network when the scene graph in Figure 2 (a) is the input. net1 is a multi-layer perceptron applied to binomial relations, which takes as input vectors (ν_i, ρ_j, ν_k) corresponding to $e = (v_i, r_j, v_k) \in E$, $(r_j \in \mathcal{R}_2)$ and outputs vectors $(\nu_{ij}, \rho'_j, \nu_{kj})$. Then, pooling and dimensionality reduction by net2 are performed for each object vector, and the first layer of the graph convolution is completed. The graph convolution network has five layers, and the output of the previous layer is used as the input to the next layer. By repeating this process, information on each object vector and relation vector is propagated along the edges.

Next, we describe the discriminators. In sg2im (Johnson et al., 2018), there are two discriminators: D_{img} and D_{obj}. D_{img} takes a generated image or a training image as input and outputs the probability that the input is a generated image. D_{obj} takes as input a generated object in the generated image or an object in the dataset image and outputs the probability that the input is a generated object and the probability that its category is $c \in \mathcal{C}$. By identifying the category of each object with D_{obj}, we can learn the semantic consistency between the word and the image.

4.2 Shortcomings of sg2im (Johnson et al., 2018)

As described in Section 1, sg2im (Johnson et al., 2018) has a shortcoming: the positional relations among three or more objects tend to be inaccurate. The edges in a scene graph can represent only relations between two objects. Therefore, a multilayer perceptron (MLP) in the graph convolution network is applied to the features of three objects in two separate steps. For example, for a path “cat” → “left of” → “dog” →“left of” → “sheep” in the scene graph, the convolution is performed on the two edges “cat” → “left of” → “dog” and “dog” → “left of” → “sheep”. This fact leads to the difficulty of sg2im in capturing the positional relations among three or more objects.

Moreover, sg2im (Johnson et al., 2018) has difficulty in generating the objects clearly due to the low resolution (64×64) of the generated images when the input scene graph involves many objects. We confirmed that simply increasing the resolution of the generated images does not work well due to the mode collapse (Gui et al., 2021), which is a phenomenon that the generated images are all similar to each other due to the training failure.

5 PROPOSED MODEL

To address the shortcoming of the existing image generation model from a scene graph, we propose an image generation model from a hyper scene graph hsg2im. In this Section, we explain the proposed model hsg2im, and training of hsg2im.

5.1 hsg2im

As explained in Section 1, sg2im (Johnson et al., 2018) has a shortcoming: the positional relations among three objects tend to be inaccurate due to the convolution method.

We address this shortcoming by modifying two parts from sg2im (Johnson et al., 2018): generating images from a hyper scene graph with a trinomial hyperedge and extending the graph convolution network to the hypergraph convolution network. Also, to generate a higher resolution image, we use a layout-to-image model Layout2img (He et al., 2021). Layout2img generates images whose resolution is 128

$\times 128$. In this Section, we explain image generation using the hypergraph convolution network and Layout2img (He et al., 2021). An overview of the generator G of hsg2im is shown in Figure 3 (b). The discriminator is unchanged from that of sg2im.

First, we create the hypergraph convolution network by adding a multi-layer perceptron net3, which is applied to trinomial hyperedges, to the graph convolution network of sg2im. The net3 takes vectors $(\nu_i, \rho_j, \nu_k, \nu_l)$ as input and outputs vectors $(\nu_{ij}$, ρ'_j, ν_{kj}, $\nu_{lj})$ corresponding to a trinomial hyperedge $q = (v_i, r_j, v_k, v_l) \in Q$. The addition of net3 enables the convolution of the hyper scene graph with trinomial hyperedges. As a result, three objects are processed with an application of net3 and two applications of net1, and their positional relation is expected to be improved. Furthermore, the improved positional relation reduces the overlapping of objects and is expected to produce more consistent layouts with the input, which leads to generating better images. Figure 4 shows the application of the hypergraph convolution network to the hyper scene graph in Figure 2 (b).

Next, we describe image generation using Layout2img (He et al., 2021). To address the image resolution shortcoming of sg2im, we use the pre-trained model Layout2img, which generates a 128×128 image from layout $L = \{(c_i, b_i)\}_{i=1}^{N}$, as an auxiliary model. $\{(c_i)\}_{i=1}^{N}$ is obtained from the input (hyper) scene graph. Bounding box $\hat{B} = \{(b_i)\}_{i=1}^{N}$ is generated in the framework of sg2im (Johnson et al., 2018) with the hypergraph convolutional neural network. Layout2img uses the structure of ResNet (He et al., 2016) in its generator, so that even models which generate a high resolution image (128×128) can be trained stably.

5.2 Training of hsg2im

The training of generator G and discriminator D is performed alternately. During the training phase, feature map M is created from bounding box B in the dataset. Loss L_G of generator G is $L_G = \sum_{i=1}^{5} w_i L_i$, where w_i is the weight of loss L_i defined as follows.

- Loss with respect to image pixels: $L_1 = L_{\text{pix}}$
- Loss with respect to the bounding box: $L_2 = L_{\text{box}}$
- Adversarial loss from discriminator D_{img}: $L_3 = L_{\text{GAN}}^{\text{img}}$
- Adversarial loss from discriminator D_{obj}: $L_4 = L_{\text{GAN}}^{\text{obj}}$
- Auxiliarly classifier loss from D_{obj}: $L_5 = L_{\text{AC}}^{\text{obj}}$

The losses $(L_{\text{pix}}, L_{\text{box}}, L_{\text{GAN}}^{\text{img}}, L_{\text{GAN}}^{\text{obj}}, L_{\text{AC}}^{\text{obj}})$ and their weights set are the same as in (Johnson et al., 2018).

6 EXPERIMENTS

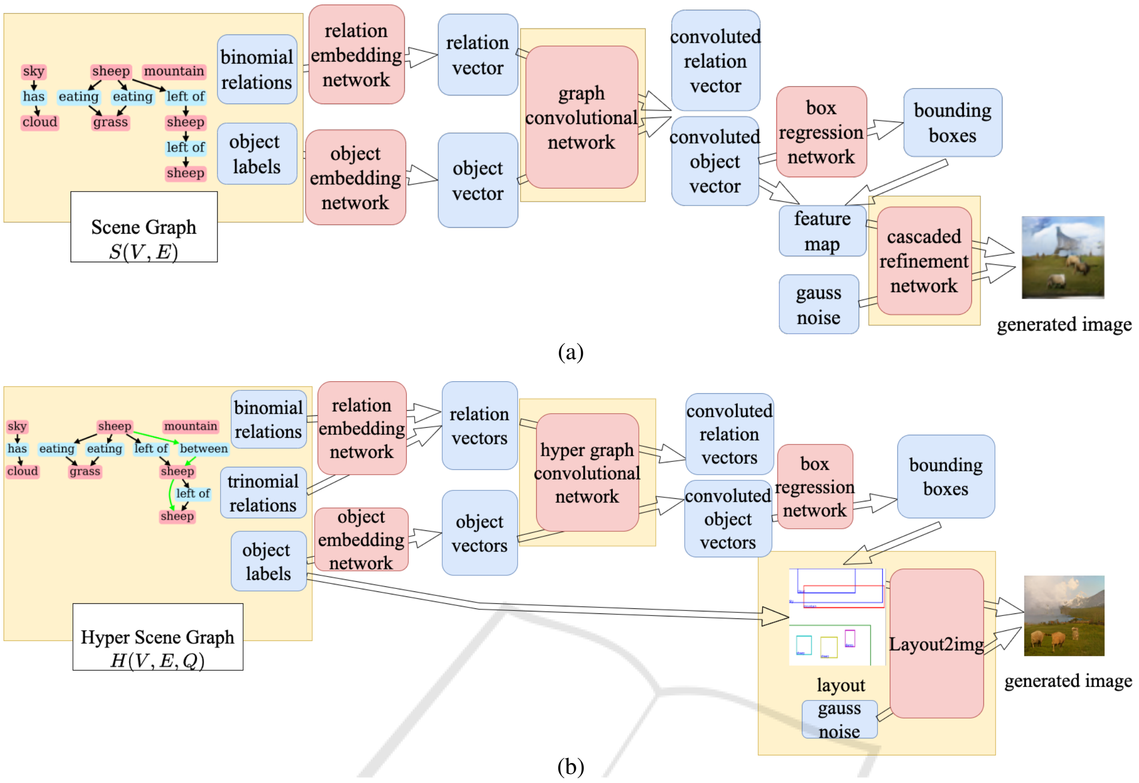
6.1 Dataset

We use as datasets the 2017 COCO Stuff (Caesar et al., 2018) and Visual Genome (Krishna et al., 2017) by adding trinomial hyperedges. The 2017 COCO-Stuff dataset (COCO) (Caesar et al., 2018) consists of images, objects, their bounding boxes, and segmentation masks. There are 40,000 training images and 5,000 validation images. Since the COCO dataset does not contain data on relations between objects, we construct a hyper scene graph by adding seven relations “left of”, “right of”, “above”, “below”, “inside”, “surrounding”, and “between” based on the bounding boxes. The “between” relation is added as a trinomial hyperedge in hsg2im. Since there is no test set in the COCO dataset, we divide the validation set into a new validation set and a test set. As a result, the COCO dataset contains 24,972 training images, 1,667 validation images, and 3,333 test images.

The Visual Genome (Krishna et al., 2017) version 1.4 (VG) dataset, consisting of 108,077 images annotated with scene graphs, consists of images, objects, their bounding boxes, and the binomial relations between the objects. In the entire dataset, only objects which appear more than 2,000 times and relations which appear more than 500 times are used following (Johnson et al., 2018) in our experiments. Samples with less than 2 objects and more than 31 objects are ignored. As a result, we use 62,602 training images, 5,069 evaluation images, and 5,110 test images. For Visual Genome, we add the “between” relation (4,136 hyperedges) as in COCO.

In addition, to compare sg2im with our hsg2im, we also add the binomial edges “left of” (8,272 edges) corresponding to hyperedges “between” in both datasets: $(v_i$, “left of”, $v_j)$ and $(v_j$, “left of”, $v_k)$, together which correspond to $(v_i$, “between”, $v_j, v_k)$.

Next, we explain how to add hyperedges “between” and “left of”. For hsg2im, both a trinomial hyperedge $(v_i$, “between”, $v_j, v_k)$ and two binomial edges $(v_i$, “left of”, $v_j)$ and $(v_i$, “left of”, $v_j)$ are added for three objects $(v_i, b_i), (v_j, b_j), (v_k, b_k)$ which satisfy all of the five conditions in Section 3.3. For sg2im, only the latter two are added. In Visual Genome, we add 3,477 hyperedges, 334 hyperedges, and 325 hyperedges for the training dataset, the validation dataset, and the test dataset, respectively. In COCO, we add 9,437 hyperedges, 144 hyperedges, and 227 hyperedges for the training dataset, the validation dataset, and the test dataset, respectively.



(a)

(b)

Figure 3: Example of generating an image not using Layout2img (He et al., 2021) with generator G of sg2im (Johnson et al., 2018) (a) and using Layout2img with generator G of hsg2im (b). The elements that we modified are highlighted in yellow.

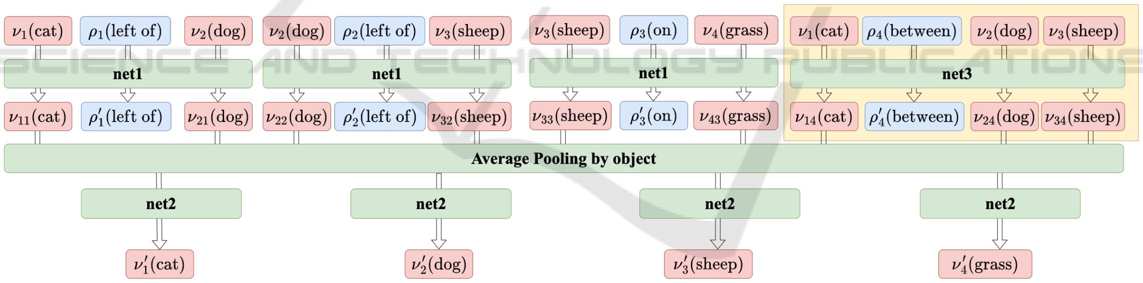


Figure 4: Flowchart of one of the five layers of the hyper graph convolutional network of hsg2im. The part without the yellow region shows that of the graph convolutional network of sg2im (Johnson et al., 2018). net1, net2, and net3 represent MLPs for binomial edges, dimension reduction, and trinomial hyperedges, respectively. The detailed structures of net1, net2, and net3 are given in APPENDIX.

6.2 Experimental Conditions

We train sg2im and hsg2im on the COCO and Visual Genome datasets, respectively, based on the method described in Section 5.2. Here, the learning algorithm (Adam (Kingma and Ba, 2015)), the learning rate ($= 10^{-4}$), the batch size ($= 32$), and the iteration upper bound ($= 10^6$) are the same as in the sg2im paper (Johnson et al., 2018).

6.3 Results and Discussion

First, we quantitatively evaluate hsg2im. The evaluation in terms of PTO is shown in Table 1. We see that hsg2im shows 12% and 20% higher scores than s2im on the COCO and VG datasets, respectively. These results are because the trinomial hyperedge enables us to capture the positional relations among three objects.

We show the evaluation in terms of AoO in Table 2. hsg2im shows a higher score on the VG dataset.

Table 1: Comparison of PTO scores between sg2im and hsg2im.

	COCO	VG
sg2im	0.52	0.19
hsg2im	**0.64**	**0.39**

In the COCO dataset, the scores of sg2im and hsg2im are the same. We speculate that the identical scores on the COCO dataset are due to the few diversity of binomial edges on this dataset; sg2im generated fewer overlapping bounding boxes only with binomial relations. Therefore, the room for improvement is small. For VG, we can confirm that the addition of a trinomial hyperedge improves the positional relation of three objects and reduces the overlapping between them.

Table 2: Comparison of AoO scores between sg2im and hsg2im.

	COCO	VG
sg2im	**0.02**	0.11
hsg2im	**0.02**	**0.06**

We show the evaluation in terms of Inception Score in Table 3. In the image generated without Layout2img, hsg2im shows a 0.33 lower score for COCO and a 0.40 higher score than sg2im for Visual Genome datasets. For images generated using Layout2img, hsg2im shows 0.26 and 0.10 higher scores than sg2im for COCO and Visual Genome datasets, respectively. In the case of not using Layout2img for COCO, the reason of the degraded IS would be attributed to the lowness of the resolution. Overall, the scores show that the images generated with Layout2img are more natural than those generated by sg2im and more natural images are generated from layouts produced by hsg2im than by sg2im.

Table 3: Comparison of Inception Score between sg2im and hsg2im, and with and without Layout2img.

	COCO	VG
sg2im	4.17	5.45
hsg2im wo Layout2img	3.84	5.85
sg2im + Layout2img	11.63	9.83
hsg2im + Layout2img	**11.89**	**9.93**

Next, we qualitatively evaluate the generated images. Figure 5 shows the generated images of hsg2im and sg2im. The scene graph of the input to sg2im in (a) has a path “sheep” (light blue bounding box) → “left of” → “sheep” (yellow) → “left of” → “sheep” (purple). However, we can confirm that “sheep” (yellow) is generated at a higher position than “sheep” (light blue), and the “left of” relation is not satisfied. On the other hand, in the layout generated by hsg2im, three objects are generated in a positional relation which satisfies the trinomial relation “sheep” (light blue) → “between” → “sheep” (yellow) → “sheep” (purple) in the hyper scene graph of the input. Also, the scene graph of the input to sg2im in (b) has a path “picture” (purple) → “left of” → “light” (light blue) → “left of” → “shelf” (blue). However, “light” and “shelf” bounding boxes are generated on top of each other, and these objects are missing in the image (with Layout2img). On the other hand, in the layout generated by hsg2im, the bounding boxes for “light” and “shelf” do not overlap each other, and these objects can be seen in the generated image (with Layout2img). Thus, we can also confirm the effect of the introduction of the trinomial hyperedge in the generated images.

6.4 Computational Time

We explain the computational time of the training and the test phases in each of sg2im and hsg2im. We used one GPU (NVIDIA TITAN RTX). From Table 4, we can confirm that there is no significant difference between sg2im and hsg2im in both the training time and the test time. In some cases such as the training time in Visual Genome and COCO, hsg2im takes longer time due to the addition of net3 in the proposed model.

Table 4: Computational time of the training and the test phases in each of sg2im and hsg2im.

	Training	Test
sg2im (COCO)	66.1h	127.6m
hsg2im (COCO)	66.4h	127.5m
sg2im (VG)	31.8h	196.1m
hsg2im (VG)	34.8h	191.5m

7 CONCLUSIONS

We have proposed an image generation model hsg2im from a hyper scene graph. The proposed model is an extension of the image generation model sg2im (Johnson et al., 2018) from a scene graph with our addition of a hyperedge representing a trinomial relation to the scene graph, in order to improve the positional relations of the generated objects.

In the evaluation of the Positional relation of Three Objects (PTO), the performance of hsg2im was higher by about 12% and 20% on the 2017 COCO Stuff (COCO) dataset and the Visual Genome dataset, respectively, than sg2im. In the evaluation of the naturalness of the generated images based on the Inception Score, the scores improved by about 0.25 for

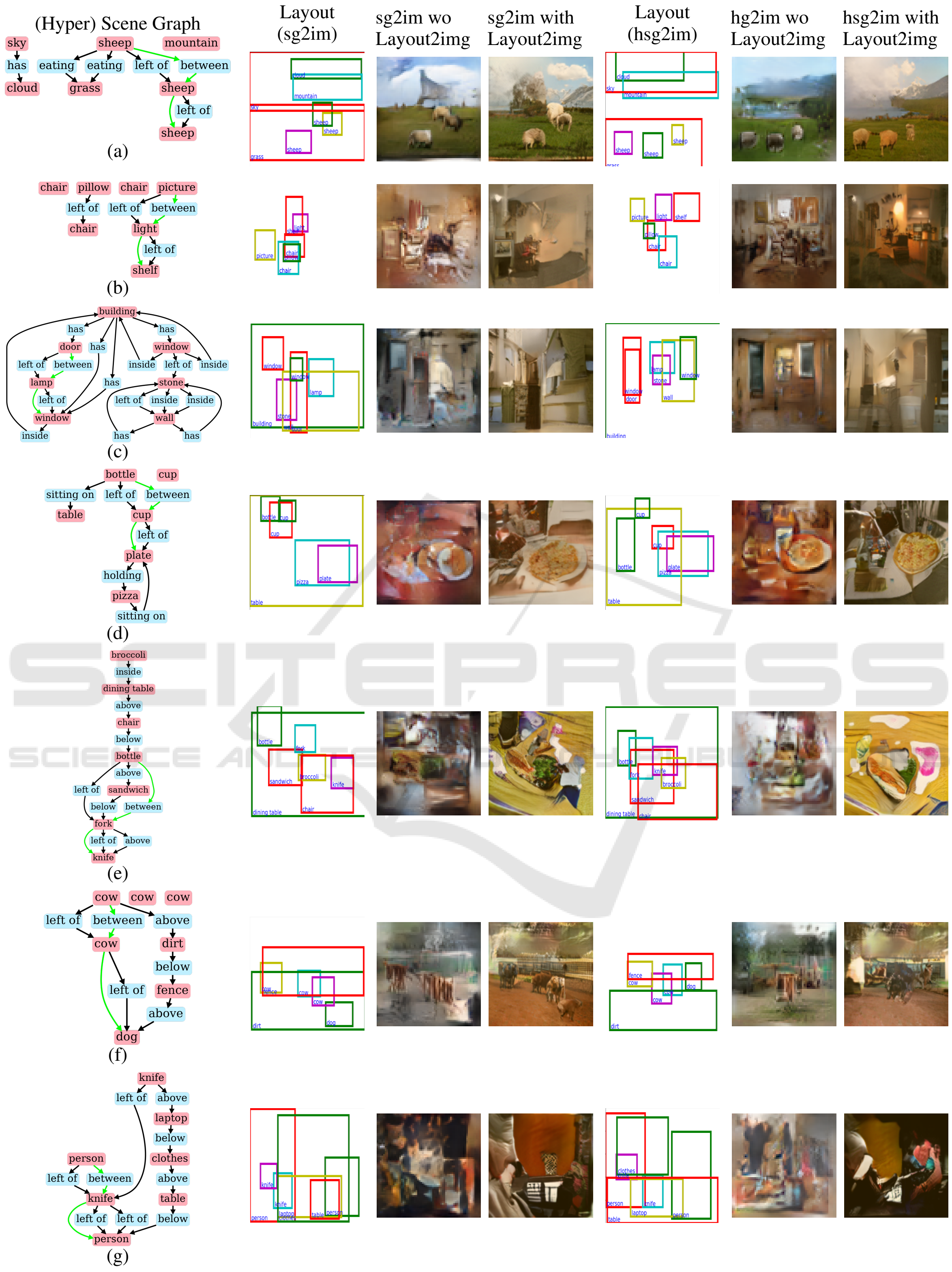


Figure 5: Comparison of images generated by sg2im (Johnson et al., 2018) and hsg2im. (a) ∼ (d) and (e) ∼ (g) are from the Visual Genome and the MS COCO datasets, respectively. In the input for sg2im, the trinomial hyperedges “between” do not exist.

the COCO dataset and by about 0.40 for the Visual Genome dataset. In addition, we generated more natural images by using the layout-to-image model Layout2img supplementally. Although we added only "between" relation as a trinomial hyperedge, adding other kinds of hyperedges would lead to generating more consistent images with the input. Thus it is an interesting direction for our future work.

REFERENCES

Agnese, J., Herrera, J., Tao, H., and Zhu, X. (2020). A Survey and Taxonomy of Adversarial Neural Networks for Text-to-Image Synthesis. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(4):e1345.

Caesar, H., Uijlings, J., and Ferrari, V. (2018). Coco-Stuff: Thing and Stuff Classes in Context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1209–1218.

Chen, Q. and Koltun, V. (2017). Photographic Image Synthesis with Cascaded Refinement Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1511–1520.

Elgammal, A., Liu, B., Elhoseiny, M., and Mazzone, M. (2017). CAN: Creative Adversarial Networks Generating "Art" by Learning About Styles and Deviating from Style Norms. *arXiv preprint arXiv:1706.07068*.

Frolov, S., Hinz, T., Raue, F., Hees, J., and Dengel, A. (2021). Adversarial Text-to-Image Synthesis: A review. *arXiv preprint arXiv:2101.09983*.

Ghorbani, A., Natarajan, V., Coz, D., and Liu, Y. (2020). DermGAN: Synthetic Generation of Clinical Skin Images with Pathology. In *Machine Learning for Health Workshop*, pages 155–170. PMLR.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Nets. *Advances in Neural Information Processing Systems*, 27.

Gui, J., Sun, Z., Wen, Y., Tao, D., and Ye, J. (2021). A Review on Generative Adversarial Networks: Algorithms, Theory, and Applications. *IEEE Transactions on Knowledge and Data Engineering*.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778.

He, S., Liao, W., Yang, M. Y., Yang, Y., Song, Y.-Z., Rosenhahn, B., and Xiang, T. (2021). Context-Aware Layout to Image Generation with Enhanced Object Appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15049–15058.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. (2017). GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *NuerIPS*.

Hinz, T., Heinrich, S., and Wermter, S. (2019). Generating Multiple Objects at Spatially Distinct Locations. *ArXiv*, abs/1901.00686.

Hinz, T., Heinrich, S., and Wermter, S. (2022). Semantic Object Accuracy for Generative Text-to-Image Synthesis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44:1552–1565.

Johnson, J., Gupta, A., and Fei-Fei, L. (2018). Image Generation from Scene Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1219–1228.

Johnson, J., Krishna, R., Stark, M., Li, L.-J., Shamma, D., Bernstein, M., and Fei-Fei, L. (2015). Image Retrieval Using Scene Graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3668–3678.

Kingma, D. P. and Ba, J. (2015). Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6980.

Kingma, D. P. and Welling, M. (2013). Auto-Encoding Variational Bayes. *arXiv preprint arXiv:1312.6114*.

Krishna, R., Zhu, Y., Groth, O., Johnson, J., Hata, K., Kravitz, J., Chen, S., Kalantidis, Y., Li, L.-J., Shamma, D. A., et al. (2017). Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*, 123(1):32–73.

Li, Y., Ma, T., Bai, Y., Duan, N., Wei, S., and Wang, X. (2019). PasteGAN: A Semi-Parametric Method to Generate Image from Scene Graph. *ArXiv*, abs/1905.01608.

Mirza, M. and Osindero, S. (2014). Conditional Generative Adversarial Nets. *arXiv preprint arXiv:1411.1784*.

Mittal, G., Agrawal, S., Agarwal, A., Mehta, S., and Marwah, T. (2019). Interactive Image Generation Using Scene Graphs. *ArXiv*, abs/1905.03743.

Nie, D., Trullo, R., Lian, J., Petitjean, C., Ruan, S., Wang, Q., and Shen, D. (2017). Medical Image Synthesis with Context-Aware Generative Adversarial Networks. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 417–425. Springer.

Odena, A., Olah, C., and Shlens, J. (2017). Conditional Image Synthesis with Auxiliary Classifier GANs. In *International Conference on Machine Learning*, pages 2642–2651. PMLR.

Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. *arXiv preprint arXiv:1511.06434*.

Reed, S. E., Akata, Z., Yan, X., Logeswaran, L., Schiele, B., and Lee, H. (2016). Generative Adversarial Text to Image Synthesis. *ArXiv*, abs/1605.05396.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., and Savarese, S. (2019). Generalized Intersection over Union: A Metric and a Loss for Bounding Box Regression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 658–666.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252.

Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., and Chen, X. (2016). Improved Techniques for Training GANs. *Advances in Neural Information Processing Systems*, 29.

Schuster, S., Krishna, R., Chang, A., Fei-Fei, L., and Manning, C. D. (2015). Generating Semantically Precise Scene Graphs from Textual Descriptions for Improved Image Retrieval. In *Proceedings of the Fourth Workshop on Vision and Language*, pages 70–80.

Sun, W. and Wu, T. (2019). Image Synthesis From Reconfigurable Layout and Style . In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10531–10540.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. (2015). Going Deeper with Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9.

Van Den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. (2016). Pixel Recurrent Neural Networks. In *International Conference on Machine Learning*, pages 1747–1756. PMLR.

Vo, D. M. and Sugimoto, A. (2020). Visual-Relation Conscious Image Generation from Structured-Text. *ArXiv*, abs/1908.01741.

Wu, X., Xu, K., and Hall, P. (2017). A Survey of Image Synthesis and Editing with Generative Adversarial Networks. *Tsinghua Science and Technology*, 22(6):660–674.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. (2017). StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5907–5915.

Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., and Metaxas, D. N. (2018). Stackgan++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(8):1947–1962.

APPENDIX

We show the detailed structure of hg2sim. Fig. 6,7, and 8 are net1, net2 and net3, which are used in the (hyper) graph convolutional network, respectively.

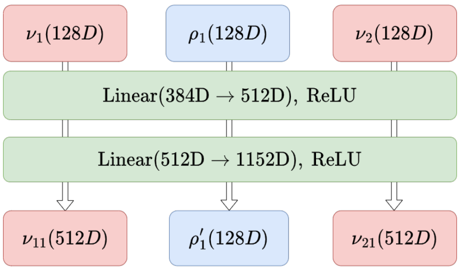


Figure 6: Structure of net1. It receives two 128-dimensional object vectors and one relation vector corresponding to $e \in E$, and outputs two 512-dimensional object vectors and one 128-dimensional relation vector.

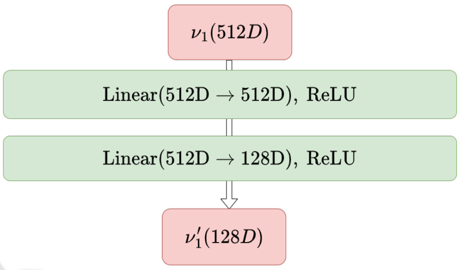


Figure 7: Structure of net2. It receives one 512-dimensional object vector, which is the output of net1 and net3, and outputs a 128-dimensional object vector by conducting dimensionality reduction.

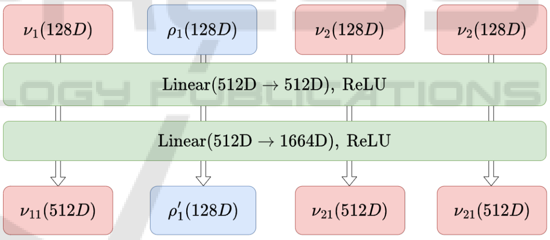


Figure 8: Structure of net3. It receives three 128-dimensional object vectors and one relation vector corresponding to $q \in Q$, and outputs three 512-dimensional object vectors and one 128-dimensional relation vector.