Text as Neural Operator: Complex Image Manipulation by Text Instruction

Tianhao Zhang* **Hung-Yu Tseng** **Lu Jiang** **Honglak Lee** **Irfan Essa** **Weilong Yang**
Google Research
Mountain View, CA, USA
<bryanzhang,hungyutseng,lujiang,honglak,irfanessa,weilongyang>@google.com

Abstract

In recent years, text-guided image manipulation has gained increasing attention in the image generation research field. Recent works have proposed to deal with a simplified setting where the input image only has a single object and the text modification is acquired by swapping image captions or labels. In this paper, we study a setting that allows users to edit an image with multiple objects using complex text instructions. In this image generation task, the inputs are a reference image and an instruction in natural language that describes desired modifications to the input image. We propose a GAN-based method to tackle this problem. The key idea is to treat text as neural operators to locally modify the image feature. We show that the proposed model performs favorably against recent baselines on three public datasets.

1 Introduction

Image synthesis from text has been a highly active research area. This task is typically set up as a conditional image generation problem where a Generative Adversarial Network (GAN) [10] is learned to generate realistic images according to the text description in the format of natural languages [52, 50, 58, 29], scene graphs [20, 51], or other modalities [31, 38, 27].

In this paper, we study *how to manipulate image content through complex text instruction*. In this setting, a user is able to apply various changes to a reference image to add, remove, or modify its content by sending text instructions. For example, Figure 1 shows the generated images by our model for three instructions: 1) adding a new object at a location, 2) removing an object, and 3) changing the object's attributes (size, shape, color, etc).

The closest related problem to ours is text-guided image manipulation [38, 27] which demonstrates promising image manipulation quality from the text. Sequential text-to-image generation known as GeNeVA [8], which focuses on sequentially adding objects to a blank canvas given step-by-step text instructions, is also related, as each step can be seen as doing text-guided image manipulation on the intermediate image. However, the language in existing works is limited in complexity and diversity which consists of either descriptive attributes [38, 27] or a single "add" operation [8]. Different from previous works, this paper focuses on modeling the *complex instruction* for image manipulation. The studied text instructions involve adjectives (attributes), verbs (actions) and adverbs (locations) for three representative operations "add", "modify", and "remove". In addition, the complex instruction often specifies changes to only one of the many objects in the reference image as opposed to the single salient object in prior works [38, 27].

Image manipulation by complex instruction is inspired by cross-modal image retrieval which comprises a variety of applications such as product search [23, 55, 11]. In this retrieval setting [48], users search an image database using an input query that is formed of an image plus some text that

[*] Work done as a Google AI Resident.

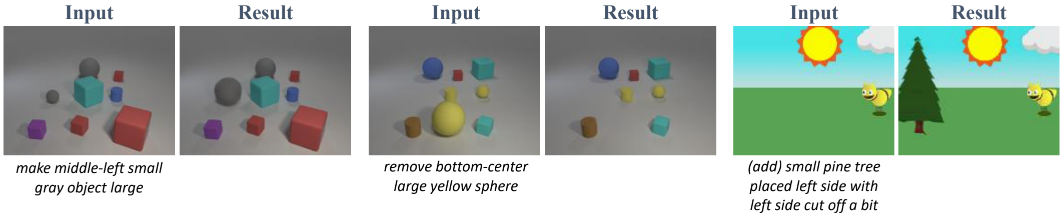

Input
Result
Input
Result
Input
Result
make middle-left small gray object large
remove bottom-center large yellow sphere
(add) small pine tree placed left side with left side cut off a bit

Figure 1: **Image manipulation by text instruction.** Each input contains a reference image and a text instruction. The results are synthesized images by our model.

describes complex modifications to the input image. Cross-modal retrieval is essentially the same as our problem except it aims at retrieving as opposed to generating the target image. Interestingly, as we will show, the generated image can be used to retrieve target images with competitive accuracy, providing a more explainable search experience that allows users to inspect the result before the retrieval.

The main research question studied in this paper is how to model the *complex text instructions* for effective conditional image manipulation. To this end, we propose an approach called Text-Instructed Manipulation GAN or TIM-GAN. The key idea is to treat language as *neural operators* to modify the image feature in a way such that the modified feature is useful in synthesizing the target image by the GAN model. The generation process is decomposed into where and how to edit the image feature. For "where to edit", we leverage existing attention mechanisms to ground words to a spatial region in the image. Although the use of spatial attention is not new, we find it allows for learning generic neural operators that can be decoupled from specific locations. For "how to edit", we introduce a novel text-adaptive routing network to generate text operators for complex instructions. For a text instruction, a route is dynamically created serving as a neural operator to modify the image feature. Since similar instructions perform similar operations, the text-adaptive routing network allows neural blocks to be shared among similar instructions, while still being able to distinguish among different operations.

Experimental results on three datasets, including Clevr [48], Abstract scene [59], and Cityscapes [6] demonstrate that image manipulation by the proposed approach outperforms baseline approaches by large margins in terms of Fréchet Inception Distance (FID) [13] and Retrieval Scores [50]. The user study validates our method's efficacy in generating more realistic and semantic relevant images. We also conduct ablation studies to substantiate the performance gain stems from the proposed neural operators.

2 Related Work

Conditional generative adversarial networks. Generative adversarial networks GANs [10, 36, 2, 3] have made rapid progress on image generation in recent years. Built on the basis of GANs, the *conditional* GAN aims to synthesize the image according to the input context. The input context can be images [18, 56, 24, 16, 37], audio sequences [25], human poses [34], or semantic segmentation masks [49, 41, 28]. Particularly, text-to-image synthesis [52, 20, 50, 58, 29, 51, 26] learns a mapping from textual descriptions to images. Recently, GeNeVA [8] extended the mapping for iterative image generation in which new objects are added one-by-one to a blank canvas following textual descriptions. Different from text-to-image synthesis, the proposed problem takes multimodal inputs, aiming at learning to *manipulate* image content through text instructions.

Conditional image manipulation. The goal is to manipulate image without degrading the quality of the edited images. To enable user-guided manipulation, a variety of frameworks [53, 54, 15, 32, 17, 43, 4, 38, 27] have been proposed to use different control signals. For instance, Zhang et al. [54] uses sparse dots to guide the image colorization process. There are additional works on image manipulation by bounding boxes subsequently refined as semantic masks [14] or by code [35]. Numerous image stylization [15, 32] and blending [17] approaches augment the images by referencing an exemplar image. Closest to ours are the TA-GAN [38] and ManiGAN [27] schemes that take the image caption as input to describe attributes for conditional image manipulation. In this work, we propose to manipulate the images according to complex text *instructions*. Different from the image caption used

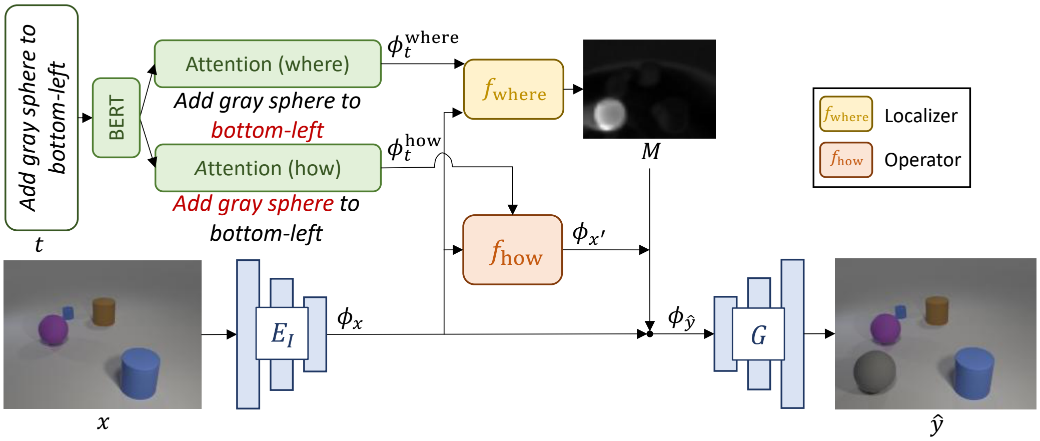


Figure 2: **Method overview.** Given an input image x and a text instruction t, the proposed TIM-GAN first predicts a spatial attention mask M (*where* to edit, Section 3.2) and a text operator f_{how} (*how* to edit, Section 3.1). The image feature ϕ_x is then modified by the text operator f_{how} on the predicted mask M. Finally, the edited image $\hat{y}$ is synthesized from the manipulated image feature $\phi_{\hat{y}}$.

by the TA-GAN and ManiGAN methods, the instruction we take as input specifies 1) the region of the image to be edited (*where*) and 2) the type of editing to be conducted (*how*).

Feature Composition. The key idea of this work is to model text as operator. This can be seen as a feature composition function to combine the image and text features for image generation. Feature composition has been studied more extensively in other problems such as visual question answering [22, 40, 5, 33], visual reasoning [21, 45], image-to-image translation [57, 24], etc. In this work, we design a routing mechanism for image generation such that the intermediate neural blocks can be effectively shared among similar text operators. Our method is related to feature-wise modulation, a technique to modulate the features of one source by referencing those from the other. Examples of recent contributions are: text image residual gating (TIRG) [48], feature-wise linear modulation (FiLM) [42], and feature-wise gating [9]. Among numerous existing works on feature composition, this paper compares the closely related methods including a state-of-the-art feature composition method for image retrieval [48] and three strong methods for conditional image generation [58, 38, 8], in additional to the standard routing mechanism [44] in the ablation study.

3 Methodology

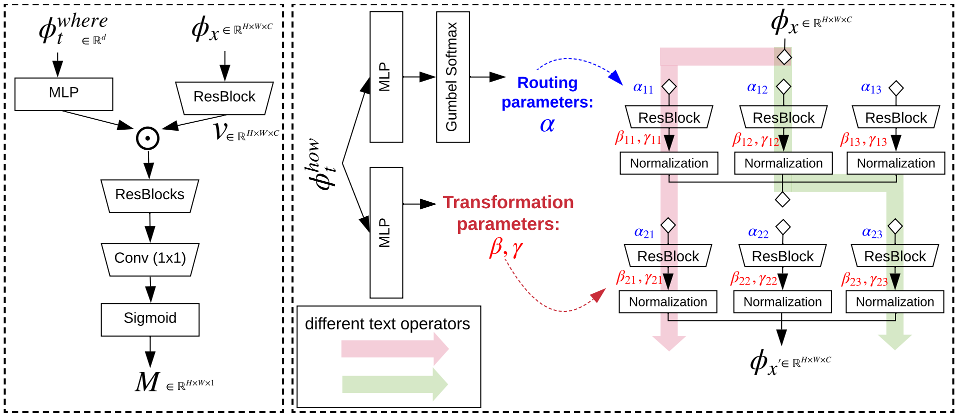
Our goal is to manipulate a given reference image according to the modification specified in the input text instruction which specifies one of the three operations "add", "modify", and "remove" We accomplish this task by modeling instructions as neural operators to specify *where* and *how* to modify the image feature.

An overview of the proposed TIM-GAN method is illustrated in Figure 2. Given the input image x and text instruction t, we first extract the image feature ϕ_x along with the text features ϕ_t^{where} and ϕ_t^{how}. The text features ϕ_t^{where} and ϕ_t^{how} encode the *where* and *how* information about the modification, respectively. To indicate the region on the image x to be edited, we predict a spatial attention mask M from ϕ_t^{where}. Thereafter, we design a new network routing mechanism for building an operator f_{how}, from the feature ϕ_t^{how}, to modulate the feature editing. Finally, the resulting image $\hat{y}$ is generated from the manipulated image feature $\phi_{\hat{y}}$ using the generator G.

Although spatial attention has been commonly used in GAN models, we find that by disentangling how from where in the modification, our model learns more generic text operators that can be applied at various locations. To be more specific, let M be a learned spatial mask. The image feature ϕ_x is modified by:

$$\phi_{\hat{y}} = (1 - M) \odot \phi_x + M \odot f_{\text{how}}(\phi_x, \phi_t^{\text{how}}; \Theta_{\text{how}}(t)), \tag{1}$$

where $\odot$ is element-wise dot product. The first term is a gated identity establishing the input image feature as a reference to the output modified feature.


$\phi_t^{where} \in \mathbb{R}^d$
$\phi_x \in \mathbb{R}^{H\times W\times C}$
MLP
ResBlock
$\odot$
$V \in \mathbb{R}^{H\times W\times C}$
ResBlocks
Conv (1x1)
Sigmoid
$M \in \mathbb{R}^{H\times W\times 1}$
ϕ_t^{how}
MLP
Gumbel Softmax
MLP
Routing parameters: α
Transformation parameters: β, γ
different text operators
$\phi_x \in \mathbb{R}^{H\times W\times C}$
α_{11}
α_{12}
α_{13}
ResBlock
ResBlock
ResBlock
β_{11}, γ_{11}
β_{12}, γ_{12}
β_{13}, γ_{13}
Normalization
Normalization
Normalization
α_{21}
α_{22}
α_{23}
ResBlock
ResBlock
ResBlock
β_{21}, γ_{21}
β_{22}, γ_{22}
β_{23}, γ_{23}
Normalization
Normalization
Normalization
$\phi_{x'} \in \mathbb{R}^{H\times W\times C}$

(a) Localizer: f_{where} **(b) Text operator: f_{how}**

Figure 3: ***Where* and *how* to edit.** (a) The calculation of spatial mask M from text feature ϕ_t^{where} and image feature ϕ_x. (b) The proposed text-adaptive routing mechanism executes various paths as text operators. The operator is parameterized by (α, β, γ) generated from text feature ϕ_t^{how}.

The second term f_{how} is the proposed neural operator function which embodies the specific computation flow over the image feature (i.e., how to modify). We introduce a new text-adaptive router to execute a sequence of neural blocks dynamically for each text instruction. A route is parameterized by $\Theta_{\text{how}}(t)$ that is generated from ϕ_t^{how}; the remaining parameters are shared across all text instructions.

For training, we use the standard conditional GAN – the pix2pix [18] model, which consists of an adversarial loss $\mathcal{L}_{\text{GAN}}$ and an ℓ_1 reconstruction loss called $\mathcal{L}_{L1}$. The weights to $\mathcal{L}_{\text{GAN}}$ and $\mathcal{L}_{L1}$ are set to 1 and 10, receptively. In the rest of this section, we will detail the computation of M and f_{how}.

3.1 How to Edit: Text-Adaptive Routing

Instructions are not independent. Similar instructions perform similar operations, e.g., "add a large cylinder" and "add a red cylinder". Motivated by this idea, we model text operators in a routing network [44] where the text feature is used to dynamically select a sequence of neural blocks (or a path). Our routing network is illustrated in Figure 3b which has l layers of m blocks of identical structures. Each block consists of a `conv` layer followed by an instance normalization layer [46]. The routing parameter α_i decides to connect or disconnect a block in a layer. An execution path is hence parameterized by a series of α for all layers.

Different from prior routing mechanisms [44, 1, 39], ours is text-adaptive which selects not only a path but also the associated parameters along the path. To be specific, in addition to α, text features also generate β and γ to perform text-specific normalization in the selected block. This design increases the learning capacity of text operators, while still allowing blocks to be shared among similar instructions. Our idea is partially inspired by the success of style transfer methods [15].

Ideally, the path selector α can only take discrete values. However, this approach is not differentiable, and continuous approximation needs to be applied. To do so, we adopt the Gumbel-Max trick [19] to sample a block from a categorical distribution. Let $\pi \in \mathbb{R}^m_{>0}$ be the categorical variable with probabilities $P(\alpha = i) \propto \pi_i$ which indicates the probability for selecting block i. We have:

$$\arg\max_i [P(\alpha = i)] = \arg\max_i [g_i + \log \pi_i] = \arg\max_i [\hat{\pi}_i], \tag{2}$$

where $g_i = -\log(-\log(u_i))$ is a re-parameterization term, and $u_i \sim \text{Uniform}(0, 1)$. To make it differentiable, the argmax operation is approximated by a continuous softmax operation: $\alpha = \text{softmax}(\hat{\pi}/\tau)$, where τ is the temperature controlling the degree of the approximation.

Then, a text operator can be parameterized by $\Theta_{\text{how}}(t)$ defined in (1) as:

$$\Theta_{\text{how}}(t) = f_{\text{MLP}}(\phi_t^{\text{where}}) = \{(\alpha_i, \beta_i, \gamma_i) | \alpha_i \in [0, 1]^m, \gamma_i, \beta_i \in \mathbb{R}^{m\times p}, i \in \{1, \cdots, l\}\}, \tag{3}$$

where the text feature ϕ_t^{where} generates real vectors $\alpha_i, \beta_i, \gamma_i$ for text-adaptive routing for all layers, p is the number of normalization parameters for each block.

Finally, as shown in Figure 3, the image feature is modified by:

$$a^{(i+1)} = \sum_{j=1}^{m} \alpha_{ij}(\gamma_{ij}\frac{o_{ij} - \mu(o_{ij})}{\delta(o_{ij})} + \beta_{ij}), \tag{4}$$

where o_{ij} is the output of the j-th `conv` block in layer i. δ and μ compute channel-wise mean and variance across spatial dimensions, and are applied at test time unchanged. The operator in (4) takes the input of $a^{(1)} = \phi_x$ and outputs the modified image feature as $a^{(l)}$.

3.2 Where to Edit: Spatial Mask

We use the standard scaled dot-product self-attention [47] to summarize the location-indicative words in an instruction. Let $S = [w_1, \cdots, w_l] \in \mathbb{R}^{l \times d_0}$ denote the instruction where $w_i \in \mathbb{R}^{d_0}$ is the BERT embedding [7] for the i-th word. The query, key and value in the attention are computed by:

$$Q = SW_Q, \qquad K = SW_K, \qquad V = SW_V \tag{5}$$

where $W_Q, W_K, W_V \in \mathbb{R}^{d_0 \times d}$ are weight matrices to learn, and d is the output dimension. After reducing matrix Q to a column vector $\hat{q}$ by average pooling along its first dimension, we obtain the attended text embedding by:

$$\phi_t^{\text{where}} = V^T \text{softmax}(\frac{K\hat{q}}{\sqrt{d}}), \tag{6}$$

in which the softmax function is supposed to assign higher attention weights for locational words. Likewise, we obtain the text feature ϕ_t^{how} for salient operational words in the instruction (e.g., “add”, “red”, “cylinder”), computed by a separate self-attention head similar to that of ϕ_t^{where}.

After that, we pass the image feature ϕ_x to a convolution block (e.g., a ResBlock [12]) to get the output $v \in \mathbb{R}^{H \times W \times C}$. The spatial mask is then computed from ϕ_t^{where} using image features as the context:

$$M = f_{\text{where}}(\phi_x, \phi_t^{\text{where}}) = \delta(W_m * (f_{\text{MLP}}(\phi_t^{\text{where}}) \odot v)) \in [0, 1]^{H \times W \times 1}, \tag{7}$$

where σ is the sigmoid function, $*$ represents the 2d-convolution product with kernel W_m (see Figure 3a). We use two layers of MLP with the ReLU activation. The spatial attention can be derived from M by performing ℓ_1 normalization over the spatial dimensions. In this paper, we choose to use the unnormalized mask for improved generalization performance.

In training, we also use an ℓ_1 loss to penalize the distance between the predicted mask M and the noisy true mask, and assign it the same weight as the $\mathcal{L}_{L1}$ reconstruction loss. Note that computing this loss needs no additional supervision as the noisy mask is automatically computed by comparing the difference between the input and ground-truth training images.

4 Experimental Results

We conduct experiments to quantitatively and qualitatively compare the proposed method with baseline approaches. Additional qualitative results are presented in the supplementary material. We will release the source code and dataset to facilitate further research in this field.

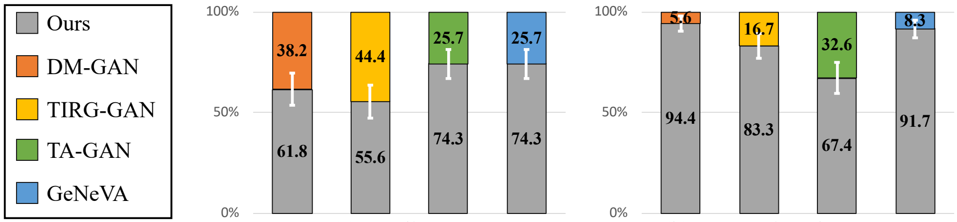
4.1 Experimental Setups

Datasets. We use three public datasets: Clevr [48], Abstract scene [59], and Cityscapes [6]. All datasets consist of images of multiple objects accompanied by complex text instructions. Since there is no dataset of text instructions on real-world RGB images (i.e., providing the ground-truth manipulated images for the inputs of a text instruction and a reference image), existing works [8, 30] were only able to test on synthetic images. Therefore we extend our method to manipulate semantic segmentation in Cityscapes. By doing so, we show the potential of our method for synthesizing RGB images from the modified segmentation mask. We describe details about these datasets in the Appendix.

Baselines. We compare to the following baseline approaches in our experiments. All methods including ours are trained and tested on the same datasets, implemented by their official code or adapted official code. More details about the baseline comparison are discussed in the Appendix.

Table 1: **Quantitative comparisons.** We use the FID scores to measure the realism of the generated images, and the retrieval score (RS) to estimate the correspondence to text instruction.

Method	Clevr			Abstract scene			Cityscape		
	FID ↓	RS@1 ↑	RS@5 ↑	FID ↓	RS@1 ↑	RS@5 ↑	FID[2] ↓	RS@1 ↑	RS@5 ↑
DM-GAN [58]	**27.9**	1.6±0.1	5.6±0.1	53.8	2.1±0.1	6.6±0.1	18.7	4.6±0.2	15.7±0.2
TIRG-GAN [48]	34.0	48.5±0.2	68.2±0.1	52.7	23.5±0.1	38.8±0.1	6.1	25.0±0.3	88.9±0.3
TA-GAN [38]	58.8	40.8±0.1	64.1±0.1	44.0	26.9±0.2	46.3±0.1	6.7	36.8±0.4	79.8±0.3
GeNeVA [8]	46.1	34.0±0.1	57.3±0.1	72.2	17.3±0.2	31.6±0.2	10.5	14.5±0.4	46.1±0.3
Ours	33.0	**95.9**±0.1	**97.8**±0.1	**35.1**	**35.4**±0.2	**58.7**±0.1	**5.9**	**77.2**±0.4	**99.9**±0.1
Real images	17.0	100	100	14.0	100	100	4.4	100	100



(a) Realism

(b) Semantic relevance

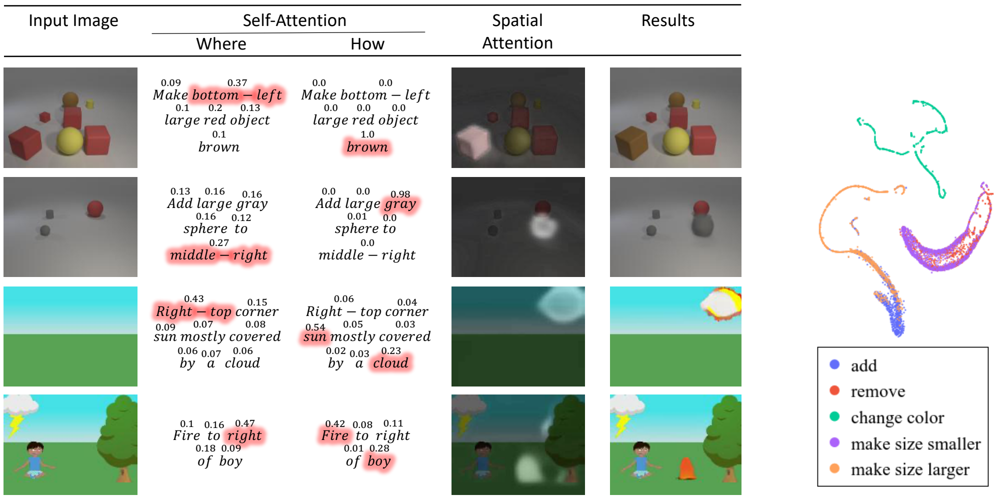
Figure 4: **User preference studies.** We present manipulated images on the Clevr and abstract scene datasets and ask the users to select the one which (a) is more *realistic* and (b) is more *semantically relevant* to the ground-truth image.

- **DM-GAN**: The DM-GAN [58] model is a recent text-to-image synthesis framework. To adapt it to our task, we use our image encoder to extract the image feature and concatenate it with its original text feature as its input signal.
- **TIRG-GAN**: TIRG [48] is a state-of-the-art method for the cross-modal image retrieval task. It takes the same input as ours but only produces the image feature for retrieval. We build a baseline TIRG-GAN based on TIRG by using our image decoder G to synthesize the image from the feature predicted by the TIRG model.
- **TA-GAN**: TA-GAN [38] is trained by learning the mapping between the caption and the image. The manipulation is then conducted by changing the caption of the image. Since there is no image caption in our task, we concatenate the pre-trained features of the input image and text instruction as the input caption feature for the TA-GAN model.
- **GeNeVA**: GeNeVA [8] learns to generate the image step-by-step according to the text description. To adapt it to take the same input as all the other methods, we use it for single-step generation over the real input image.

Metrics. In all experiments, we use the Fréchet Inception Distance score (FID) to measure the realism of the edited images [13], and the retrieval score (RS) to estimate the correctness of the manipulation. For the retrieval score, we employ the evaluation protocols similar to [48, 50]. Specifically, we use the generated image as a query to retrieve the target images in the test set. We extract the image features of all query and target images by an autoencoder pre-trained on each dataset and use simple cosine similarity between their feature embedding as our retrieval metric. The score RS@N indicates the recall of the ground-truth image in the top-N retrieved images. The computations of FID and RS scores are detailed in the Appendix.

4.2 Quantitative Results

Realism and Retrieval Score. The results are shown in Table 1. The proposed method performs favorably against all baseline approaches across datasets. Although DM-GAN appears to generate more realistic images on the Clevr dataset, its retrieval scores are very poor ($< 2\%$), indicating it merely memorizes random images without properly editing the input image. In comparison, our approach achieves a decent realism score as well as significantly higher retrieval scores.



(a) Text and spatial attention

(b) Text Operators

Figure 5: **Where and how to edit.** (a) We visualize the predicted self-attention weights and spatial attention masks. The self-attention weights are labeled above each word, and highlighted if the weights are greater than 0.2. (b) We show the t-SNE visualization of the routing parameters α predicted from different types of instructions on the Clevr dataset.

Table 2: **Ablation Studies.** Performance on ablated versions of our model.

Methods	f_{where}	f_{how}		Clevr			Abstract scene		
		text-adaptive	non-adaptive	FID ↓	R@1 ↑	R@5 ↑	FID ↓	R@1 ↑	R@5 ↑
Ours Full	✓	✓	✗	33.0	$95.9_{\pm 0.1}$	$97.8_{\pm 0.1}$	35.1	$35.4_{\pm 0.2}$	$58.7_{\pm 0.1}$
no f_{where}	✗	✓	✗	34.8	$81.7_{\pm 0.1}$	$89.6_{\pm 0.1}$	48.7	$28.7_{\pm 0.1}$	$44.4_{\pm 0.1}$
no f_{how}	✓	✗	✗	34.7	$49.5_{\pm 0.1}$	$67.4_{\pm 0.1}$	36.0	$33.8_{\pm 0.2}$	$56.7_{\pm 0.2}$
no text-adaptive	✓	✗	✓	45.9	$29.9_{\pm 0.2}$	$49.1_{\pm 0.1}$	37.4	$33.1_{\pm 0.2}$	$54.5_{\pm 0.1}$

User preference study. We conduct two user studies to understand the visual quality and semantic relevance of the generated content. Given a pair of images generated by two different methods, users are asked to choose 1) which one looks more *realistic* while ignoring the input image and text; 2) which one is more relevant to the text instruction by comparing the *content* of the generated and the ground-truth image. In total, we collect 960 answers from 30 users. As shown in Figure 4, the proposed TIM-GAN outperforms other methods by a large margin in both metrics.

Ablation study. Results are shown in Table 2. We verify three of our key designs by leaving the module out from the full model. (1) the learned mask M is removed and replaced with an identity matrix; (2) the f_{how} operator is substituted with a fixed network with the same number of layers and parameters that takes the input of concatenated features of image and text; (3) We examine the standard routing by treating the text-adaptive parameters β, γ as latent variables in our full model.

The results show the following. First, removing f_{how} from our approach leads to worse performance than the baseline methods in Table 1, which indicates the performance gain is primarily resulted from the proposed text operator as opposed to the network backbone or BERT embedding. Second, there is a sharp drop when text-adaptive routing is removed. This is more evident on Clevr in which text instructions are more diverse. These results substantiate the efficacy of text-adaptive in modeling complex text instructions. Finally, though f_{where} is not a crucial component, it complements the learning of generic f_{how} that is decoupled from specific spatial locations.

4.3 Qualitative Results

Qualitative results are shown in Figure 6. As shown, TA-GAN and TIRG-GAN tend to copy the reference images. DM-GAN often generates random objects following similar input layouts. GeNeVA can make local modifications to images, but often does not follow the text instructions. By comparison, our model generates images guided by the text instructions with better quality.

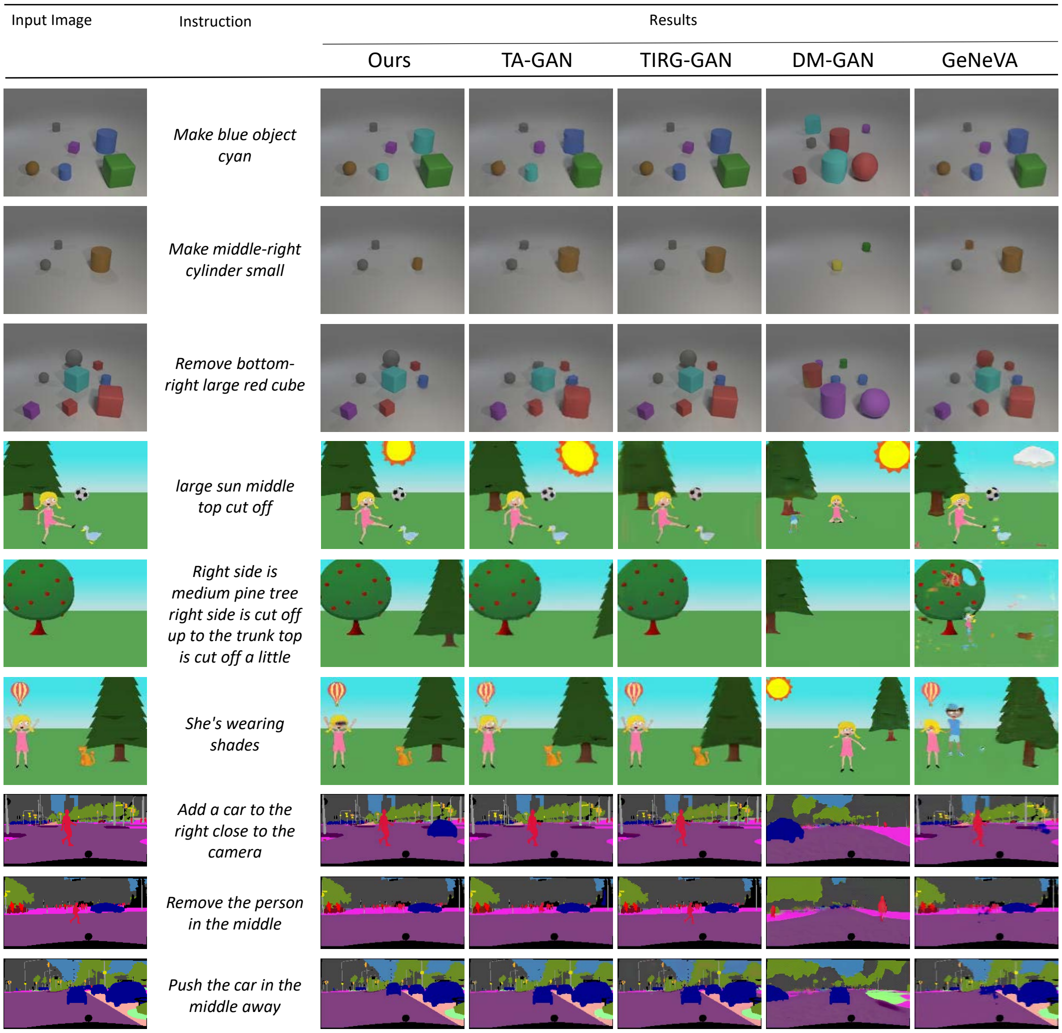

Input Image
Instruction
Results
Ours
TA-GAN
TIRG-GAN
DM-GAN
GeNeVA
Make blue object cyan
Make middle-right cylinder small
Remove bottom-right large red cube
large sun middle top cut off
Right side is medium pine tree right side is cut off up to the trunk top is cut off a little
She's wearing shades
Add a car to the right close to the camera
Remove the person in the middle
Push the car in the middle away

Figure 6: **Selected generation results.** We show the manipulation results by different approaches on the Clevr (*top*), Abstract scene (*middle*), and Cityscapes (*bottom*) datasets.

Figure 5 visualizes our intermediate results for where and how to edit. The former is shown by the text self-attention and spatial attention in Figure 5a. Figure 5b shows the t-SNE plot of the routing parameters. As shown in Figure 5b, instructions of similar types are grouped together, suggesting neural blocks are shared among similar text operators. It is interesting to find our method can automatically uncover the subtle relationship between operators, e.g., "add" and "make size larger" operators are closer indicating more neural blocks are shared between these similar operations.

5 Conclusion

In this paper, we studied a conditional image generation task that allows users to edit an input image using complex text instructions. We proposed a new approach treating text instructions as neural operators to locally modify the image feature. To learn more genetic operators, our method decomposes "where" from "how" to apply the modification (text operator), introducing a new text-adaptive network routing mechanism. We evaluate our method on three datasets and show competitive results with respect to metrics on image quality, semantic relevance, and retrieval performance.

References

[1] K. Ahmed and L. Torresani. Star-caps: Capsule networks with straight-through attentive routing. In *NeurIPS*, 2019. 4

[2] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein gan. In *ICML*, 2017. 2

[3] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 2

[4] H. Chang, J. Lu, F. Yu, and A. Finkelstein. Pairedcyclegan: Asymmetric style transfer for applying and removing makeup. In *CVPR*, 2018. 2

[5] Y. Chen, S. Gong, and L. Bazzani. Image search with text feedback by visiolinguistic attention learning. In *CVPR*, 2020. 3

[6] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 2, 5

[7] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018. 5

[8] A. El-Nouby, S. Sharma, H. Schulz, D. Hjelm, L. El Asri, S. Ebrahimi Kahou, Y. Bengio, and G. W. Taylor. Tell, draw, and repeat: Generating and modifying images based on continual linguistic instruction. In *ICCV*, 2019. 1, 2, 3, 5, 6

[9] A. Ghosh, R. Zhang, P. K. Dokania, O. Wang, A. A. Efros, P. H. Torr, and E. Shechtman. Interactive sketch & fill: Multiclass sketch-to-image translation. In *ICCV*, 2019. 3

[10] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 1, 2

[11] X. Guo, H. Wu, Y. Cheng, S. Rennie, G. Tesauro, and R. Feris. Dialog-based interactive image retrieval. In *NeurIPS*, 2018. 1

[12] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[13] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017. 2, 6

[14] S. Hong, X. Yan, T. S. Huang, and H. Lee. Learning hierarchical semantic image manipulation through structured representations. In *NeurIPS*, 2018. 2

[15] X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *ICCV*, 2017. 2, 4

[16] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 2

[17] W.-C. Hung, J. Zhang, X. Shen, Z. Lin, J.-Y. Lee, and M.-H. Yang. Learning to blend photos. In *ECCV*, 2018. 2

[18] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 2, 4

[19] E. Jang, S. Gu, and B. Poole. Categorical reparameterization with gumbel-softmax. In *ICLR*, 2017. 4

[20] J. Johnson, A. Gupta, and L. Fei-Fei. Image generation from scene graphs. In *CVPR*, 2018. 1, 2

[21] J. Johnson, B. Hariharan, L. Van Der Maaten, J. Hoffman, L. Fei-Fei, C. Lawrence Zitnick, and R. Girshick. Inferring and executing programs for visual reasoning. In *ICCV*, 2017. 3

[22] J.-H. Kim, S.-W. Lee, D. Kwak, M.-O. Heo, J. Kim, J.-W. Ha, and B.-T. Zhang. Multimodal Residual Learning for Visual QA. In *NeurIPS*, 2016. 3

[23] A. Kovashka, D. Parikh, and K. Grauman. Whittlesearch: Image search with relative attribute feedback. In *CVPR*, 2012. 1

[24] H.-Y. Lee, H.-Y. Tseng, Q. Mao, J.-B. Huang, Y.-D. Lu, M. Singh, and M.-H. Yang. Drit++: Diverse image-to-image translation via disentangled representations. *IJCV*, pages 1–16, 2020. 2, 3

[25] H.-Y. Lee, X. Yang, M.-Y. Liu, T.-C. Wang, Y.-D. Lu, M.-H. Yang, and J. Kautz. Dancing to music. In *NeurIPS*, 2019. 2

[26] B. Li, X. Qi, T. Lukasiewicz, and P. Torr. Controllable text-to-image generation. In *NeurIPS*, 2019. 2

[27] B. Li, X. Qi, T. Lukasiewicz, and P. H. Torr. Manigan: Text-guided image manipulation. In *CVPR*, 2020. 1, 2

[28] K. Li, T. Zhang, and J. Malik. Diverse image synthesis from semantic layouts via conditional imle. In *ICCV*, 2019. 2

[29] W. Li, P. Zhang, L. Zhang, Q. Huang, X. He, S. Lyu, and J. Gao. Object-driven text-to-image synthesis via adversarial training. In *CVPR*, 2019. 1, 2

[30] Y. Li, Z. Gan, Y. Shen, J. Liu, Y. Cheng, Y. Wu, L. Carin, D. Carlson, and J. Gao. Storygan: A sequential conditional gan for story visualization. In *CVPR*, 2019. 5

[31] Y. Li, L. Jiang, and M.-H. Yang. Controllable and progressive image extrapolation. *arXiv preprint arXiv:1912.11711*, 2019. 1

[32] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz. A closed-form solution to photorealistic image stylization. In *ECCV*, 2018. 2

[33] J. Liang, L. Jiang, L. Cao, Y. Kalantidis, L.-J. Li, and A. G. Hauptmann. Focal visual-text attention for memex question answering. *TPAMI*, 41(8):1893–1908, 2019. 3

[34] L. Ma, X. Jia, Q. Sun, B. Schiele, T. Tuytelaars, and L. Van Gool. Pose guided person image generation. In *NIPS*, 2017. 2

[35] J. Mao, X. Zhang, Y. Li, W. T. Freeman, J. B. Tenenbaum, and J. Wu. Program-guided image manipulators. In *ICCV*, 2019. 2

[36] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *ICCV*, 2017. 2

[37] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim. Unsupervised attention-guided image-to-image translation. In *NeurIPS*, 2018. 2

[38] S. Nam, Y. Kim, and S. J. Kim. Text-adaptive generative adversarial networks: manipulating images with natural language. In *NeurIPS*, 2018. 1, 2, 3, 6

[39] A. Newell, L. Jiang, C. Wang, L.-J. Li, and J. Deng. Feature partitioning for efficient multi-task architectures. *arXiv preprint arXiv:1908.04339*, 2019. 4

[40] H. Noh, P. Hongsuck Seo, and B. Han. Image question answering using convolutional neural network with dynamic parameter prediction. In *CVPR*, 2016. 3

[41] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu. Semantic image synthesis with spatially-adaptive normalization. In *CVPR*, 2019. 2

[42] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *AAAI*, 2018. 3

[43] T. Portenier, Q. Hu, A. Szabo, S. A. Bigdeli, P. Favaro, and M. Zwicker. Faceshop: Deep sketch-based face image editing. *ACM TOG (Proc. SIGGRAPH)*, 37(4):99, 2018. 2

[44] C. Rosenbaum, T. Klinger, and M. Riemer. Routing networks: Adaptive selection of non-linear functions for multi-task learning. In *ICLR*, 2018. 3, 4

[45] A. Santoro, D. Raposo, D. G. Barrett, M. Malinowski, R. Pascanu, P. Battaglia, and T. Lillicrap. A simple neural network module for relational reasoning. In *NIPS*, 2017. 3

[46] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In *CVPR*, 2017. 4

[47] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, 2017. 5

[48] N. Vo, L. Jiang, C. Sun, K. Murphy, L.-J. Li, L. Fei-Fei, and J. Hays. Composing text and image for image retrieval-an empirical odyssey. In *CVPR*, 2019. 1, 2, 3, 5, 6

[49] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In *CVPR*, 2018. 2

[50] T. Xu, P. Zhang, Q. Huang, H. Zhang, Z. Gan, X. Huang, and X. He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In *CVPR*, 2018. 1, 2, 6

[51] L. Yikang, T. Ma, Y. Bai, N. Duan, S. Wei, and X. Wang. Pastegan: A semi-parametric method to generate image from scene graph. In *NeurIPS*, 2019. 1, 2

[52] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *TPAMI*, 41(8):1947–1962, 2018. 1, 2

[53] R. Zhang, P. Isola, and A. A. Efros. Colorful image colorization. In *ECCV*, 2016. 2

[54] R. Zhang, J.-Y. Zhu, P. Isola, X. Geng, A. S. Lin, T. Yu, and A. A. Efros. Real-time user-guided image colorization with learned deep priors. *ACM TOG (Proc. SIGGRAPH)*, 9(4), 2017. 2

[55] B. Zhao, J. Feng, X. Wu, and S. Yan. Memory-augmented attribute manipulation networks for interactive fashion search. In *CVPR*, 2017. 1

[56] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2

[57] J.-Y. Zhu, R. Zhang, D. Pathak, T. Darrell, A. A. Efros, O. Wang, and E. Shechtman. Toward multimodal image-to-image translation. In *NIPS*, 2017. 3

[58] M. Zhu, P. Pan, W. Chen, and Y. Yang. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *CVPR*, 2019. 1, 2, 3, 6

[59] C. L. Zitnick and D. Parikh. Bringing semantics into focus using visual abstraction. In *CVPR*, 2013. 2, 5

Text as Neural Operator: Image Manipulation by Text Instruction Supplementary Materials

Anonymous Author(s)
Affiliation
Address
email

1 More Qualitative Results

1.1 More images generated by our model

We show more example images generated by our model on the three experimental datasets. See Figure 1, Figure 2, and Figure 3 for details. Generally, our model can handle complex text instructions. But we also observe two cases in which our method can fail. (a) When the location of the target is not well-specified, see Figure 4 the 8-th row. (b) When the attribute of the target is not detailed enough, see Figure 4 the 7-th and 9-th row.

1.2 Retrieval results

We use the generated image by our model as a query to retrieve the target image. Figure 4 shows the top-5 retrieved images on the Clevr dataset. We show the successful retrieval cases in the first 5 rows and failure cases in the rest of the rows.

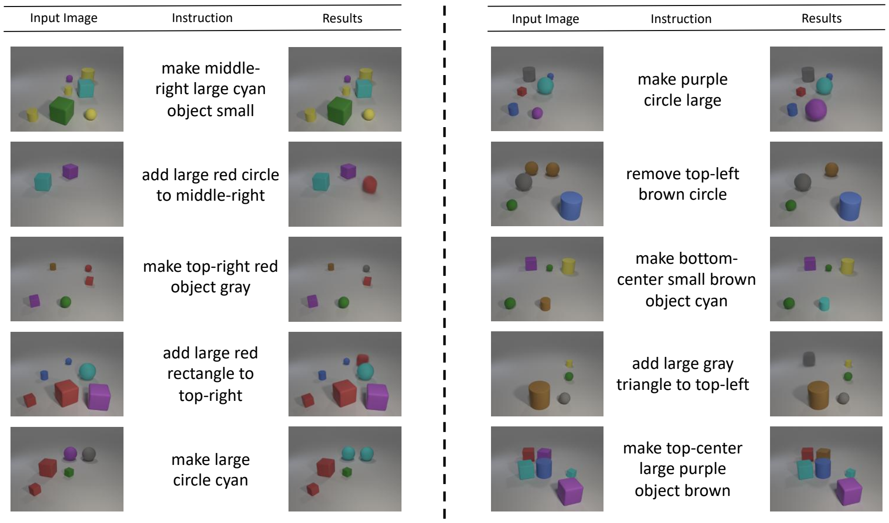

Input Image
Instruction
Results
make middle-right large cyan object small
add large red circle to middle-right
make top-right red object gray
add large red rectangle to top-right
make large circle cyan
Input Image
Instruction
Results
make purple circle large
remove top-left brown circle
make bottom-center small brown object cyan
add large gray triangle to top-left
make top-center large purple object brown

Figure 1: **Examples of the generated image by our model on Clevr.**

Submitted to 34th Conference on Neural Information Processing Systems (NeurIPS 2020). Do not distribute.

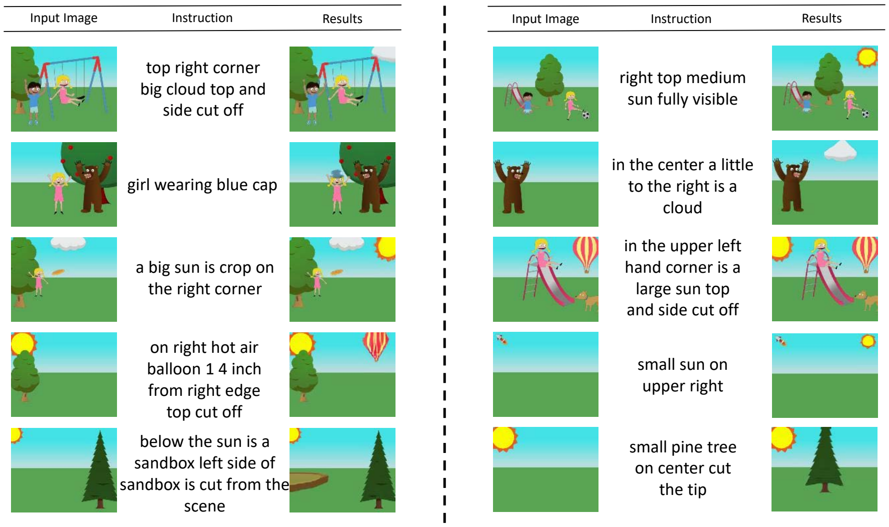

Input Image
Instruction
Results
top right corner big cloud top and side cut off
girl wearing blue cap
a big sun is crop on the right corner
on right hot air balloon 1 4 inch from right edge top cut off
below the sun is a sandbox left side of sandbox is cut from the scene
Input Image
Instruction
Results
right top medium sun fully visible
in the center a little to the right is a cloud
in the upper left hand corner is a large sun top and side cut off
small sun on upper right
small pine tree on center cut the tip

Figure 2: **Examples of the generated image by our model on Abstract Scene.**

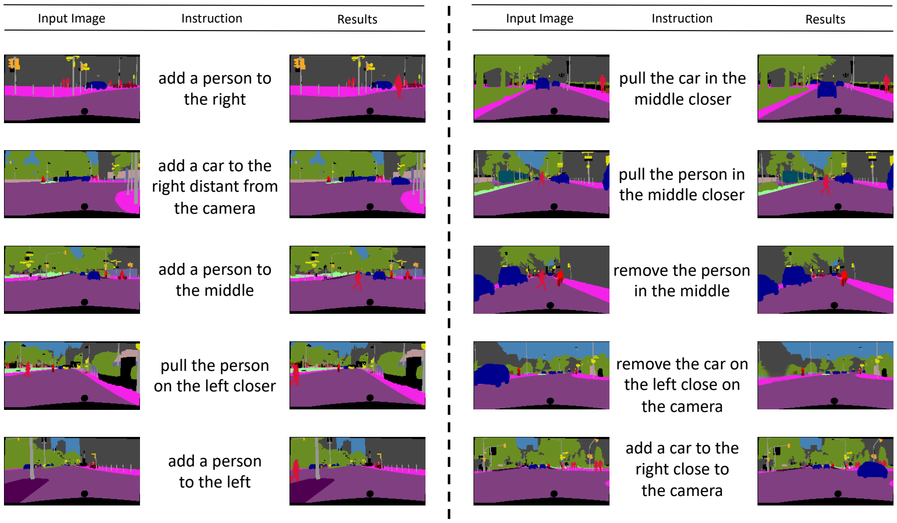

Input Image
Instruction
Results
add a person to the right
add a car to the right distant from the camera
add a person to the middle
pull the person on the left closer
add a person to the left
Input Image
Instruction
Results
pull the car in the middle closer
pull the person in the middle closer
remove the person in the middle
remove the car on the left close on the camera
add a car to the right close to the camera

Figure 3: **Examples of the generated image by our model on Cityscapes.**

2 Implementation Details

2.1 Dataset Processing

Clevr. We use the CSS dataset [**?**] which was originally created for the image retrieval task. The dataset is generated using the Clevr toolkit[**?**] and contains 3-D synthesized images with the presence of objects with different color, shape, and size. Each training sample includes an input image, an output image and a text instruction specifying the modification. There are three types of modifications: add a new object, remove an existing object, and change the attribute of an object. Each text instruction specifies the position of the target object and the expected modification. The dataset includes 17K training data pairs and 17K tests. Note that we re-render all of the images to reduce misalignment for unchanged objects.

Abstract scene. CoDraw[**?**] is a synthetic dataset built upon the Abstract Scene dataset[**?**]. It is formed by sequences of images of children playing in the park. For each sequence, there is a conversation between a Teller and a Drawer. The teller gives text instructions on how to change the current image and the Drawer can ask questions to confirm details and output images step by step. To adapt it to our setting, we extract the image and text of a single step. The dataset consists of 30K

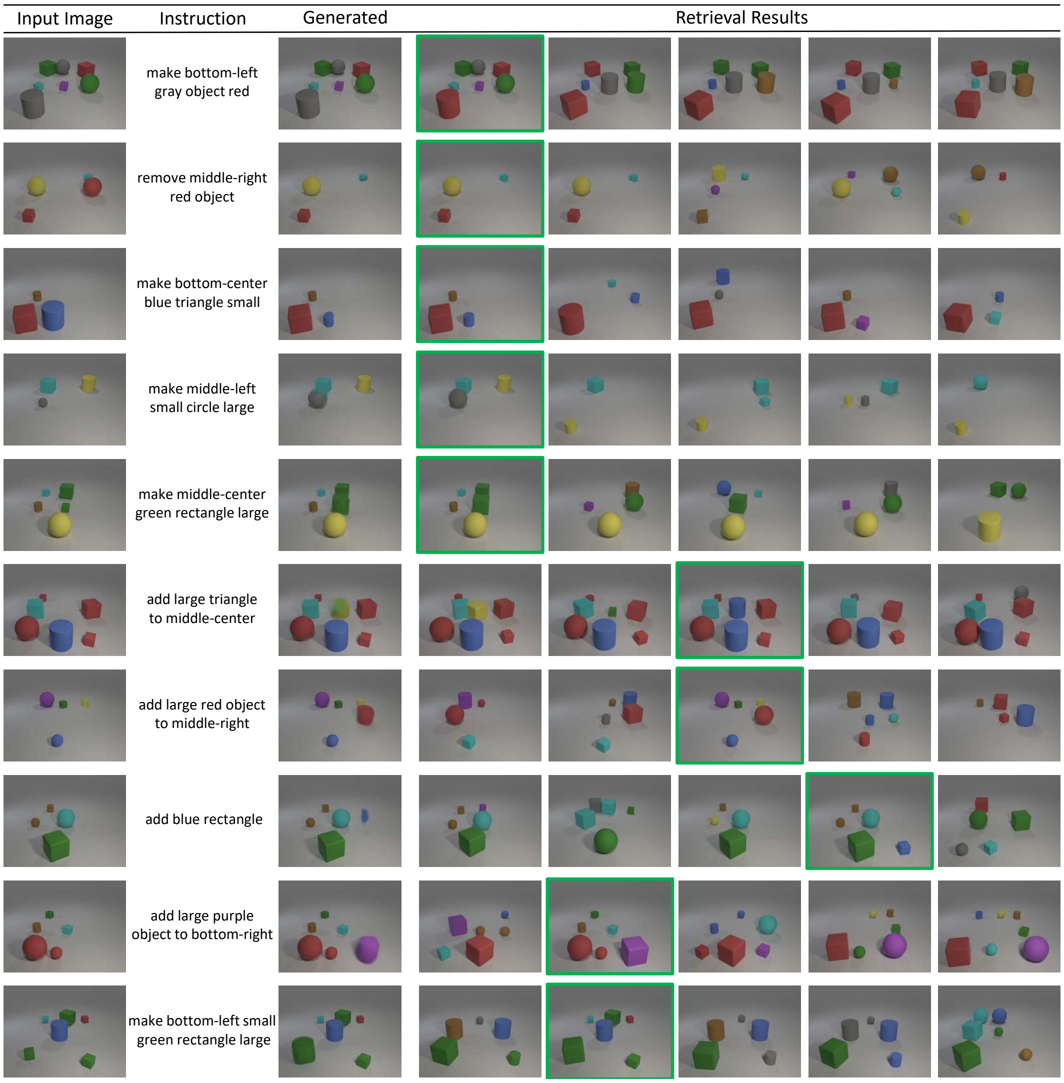

Input Image
Instruction
Generated
Retrieval Results
make bottom-left gray object red
remove middle-right red object
make bottom-center blue triangle small
make middle-left small circle large
make middle-center green rectangle large
add large triangle to middle-center
add large red object to middle-right
add blue rectangle
add large purple object to bottom-right
make bottom-left small green rectangle large

Figure 4: **Retrieval Results.** For each row, top-5 retrieved images are shown. The correct image is highlighted in the green box.

training and 8K test instances. Each training sample includes an input image, an output image and a text description about the object to be added to the input image.

Cityscapes. We propose a third dataset that is based on Cityscapes segmentation masks. The dataset consists of 4 types of text modifications: Add, Remove, Pull an object closer, and Push an object away. The ground-truth images are generated by manually pasting desired objects on the input image at appropriate positions. We crop out various object prototypes (cars, people, etc.) from existing images. Specifically, adding is done by simply pasting the added object. Removing is the inverse of adding. Pulling and pushing objects are done by pasting the same object of different sizes (with some adjustment on location as well to simulate depth changing effect). The dataset consists of 20K training instances and 3K examples for testing.

2.2 Evaluation Details

FID We adopt the standard FID [**?**] metric based on the InceptionV3 model for the Clevr dataset and the Abstract Scene dataset. However, on Cityscapes, the FID scores are computed using a pretrained auto-encoder on segmentation masks specifically. Training of the auto-encoder is exactly the same as the pretraining stage of our model. After training, we use the encoder to extract features for distance computation. We also keep the feature dimension to be the same as the original Inception V3 network to provide a similar scale of the final score.

Retrieval Score First, we extract the features of the edited images using the learned image encoder E_i to get the queries. For each query, we take the ground-truth output image and randomly select 999 real images from the test set, and extract the features of these images using the same model to form a pool for the retrieval task. Second, we compute the cosine similarity between the queries and image features from the pool. We then select the top-N most relevant images from the pool as the candidate set for each query. We report RS@1 and RS@5 scores in our experiments, in which RS@N indicates the recall of the ground-truth image in the top-N retrieved images.

2.3 Experiment Setting

We implement our model in Pytorch [**?**]. For the image encoder E_i, we use three down-sampling convolutional layers followed by Instance Normalization and ReLU activation. We use 3x3 kernels and a stride of 2 for down-sampling convolutional layers. We construct the decoder G by using two residual blocks followed by three up-sampling layers (transposed-convolutional layers) followed by Instance Normalization and ReLU activation. We use 3x3 kernels and a stride of 2 for up-sampling layers.

As for the text encoder E_t, we use the BERT [**?**] model. We use the cased version of *BERT-Base* released by the authors of the paper for the BERT [**?**] text encoder. The parameters are initialized by pretraining on a large corpus (Wikipedia + BookCorpus).

The parameters in the image encoder E_i and decoder G are initialized by training an image autoencoder. Specifically, for each dataset, we pre-train the image encoder and decoder on all images of the dataset. After the initialization, we fix the parameters in the image encoder E_i and optimize the other parts of the network in the end-to-end training. During pretraining of the autoencoder, we use the Adam optimizer [**?**] with a batch size of 8, a learning rate of 0.002, and exponential rates of $(\beta_1, \beta_2) = (0.5, 0.999)$ and train the model for 30 epochs.

In practice, the ground-truth attention mask used for the attention loss is generated by taking the absolute difference between the input and output image feature and normalizing it to a single-channel map scale from 0 to 1. The normalization consists of taking the average over the channels and dividing it by the max value.

The encoded image feature has 256 channels. The BERT output text embedding dimension $d_0 = 768$, and the attended text embedding dimension $d = 512$. The routing network has $l = 2$ layers and $m = 3$ blocks for each layer.

For the training, we use the Adam optimizer [**?**] with a batch size of 16, a learning rate of 0.002, and exponential rates of $(\beta_1, \beta_2) = (0.5, 0.999)$. We use a smaller learning rate of 0.0002 for BERT as suggested in [**?**]. The model is trained for 60 epochs. We use the loss objectives in the LSGAN [**?**] approach, and set the weights for the loss terms as follows: $\lambda_{\text{GAN}} = 1$, $\lambda_1^{\text{img}} = 10$, $\lambda_1^{\text{attn}} = 10$, $\lambda_1^{\text{feat}} = 10$.

3 Notes on Baseline Models

The selected baselines are state-of-the-art methods in text-to-image synthesis (DM-GAN[1] [**?**]), iterative text-to-image synthesis (GeNeVA[2] [**?**]), cross-modal retrieval (TIRG[3] [**?**]), attribute-based text-guided image manipulation (TA-GAN[4] [**?**]) respectively.

In addition, there is a very recent work called Mani-GAN [**?**] that also follows the setting of attribute-based image manipulation. We found the Open Access version of this paper was available in May at `http://openaccess.thecvf.com/CVPR2020.py` and we are still working on obtaining a meaningful baseline for our task.

[1] code available on `https://github.com/MinfengZhu/DM-GAN`

[2] code available on `https://github.com/Maluuba/GeNeVA`

[3] code available on `https://github.com/google/tirg`

[4] code available on `https://github.com/woozzu/tagan`

DM-GAN is originally used for text-to-image synthesis and hence there is no image input. To adapt it to our task, we add an image encoder to the model and concatenate the image feature and the text feature as the model input. However, to minimize modification on the architecture, the image feature is squeezed into a vector by using global average pooling. Therefore, significant spatial information of the input image is lost, resulting in low consistency between the generated image and the input image.