Multimedia Tools and Applications (2021) 80:27423–27443
https://doi.org/10.1007/s11042-021-11038-0

Improving text-to-image generation with object layout guidance

Jezia Zakraoui[1] · Moutaz Saleh[1] · Somaya Al-Maadeed[1] · Jihad Mohammed Jaam[1]

Received: 22 July 2020 / Revised: 22 February 2021 / Accepted: 5 May 2021 /
Published online: 20 May 2021

Abstract
The automatic generation of realistic images directly from a story text is a very challenging problem, as it cannot be addressed using a single image generation approach due mainly to the semantic complexity of the story text constituents. In this work, we propose a new approach that decomposes the task of story visualization into three phases: semantic text understanding, object layout prediction, and image generation and refinement. We start by simplifying the text using a scene graph triple notation that encodes semantic relationships between the story objects. We then introduce an object layout module to capture the features of these objects from the corresponding scene graph. Specifically, the object layout module aggregates individual object features from the scene graph as well as averaged or likelihood object features generated by a graph convolutional neural network. All these features are concatenated to form semantic triples that are then provided to the image generation framework. For the image generation phase, we adopt a scene graph image generation framework as stage-I, which is refined using a StackGAN as stage-II conditioned on the object layout module and the generated output image from stage-I. Our approach renders object details in high-resolution images while keeping the image structure consistent with the input text. To evaluate the performance of our approach, we use the COCO dataset and compare it with three baseline approaches, namely, sg2im, StackGAN and AttnGAN, in terms of image quality and user evaluation. According to the obtained assessment results, our object layout guidance-based approach significantly outperforms the abovementioned baseline approaches in terms of the accuracy of semantic matching and realism of the generated images representing the story text sentences.

Keywords Image generation · Text processing · Scene graph · Object layout · Conditioning augmentation · StackGAN

✉ Moutaz Saleh
moutaz.saleh@qu.edu.qa

[1] Department of Computer Science and Engineering, Qatar University, 2713 Doha, Qatar

1 Introduction

Image generation for the task of story visualization aims to generate meaningful and coherent images representing the story text [16]. This is a challenging task since it requires a deep understanding of the objects involved in the story as well as their mutual interactions and semantical connections [17]. Specifically, story visualization aims at generating a sequence of images to convey a given story text. The text often narrates complicated interactions with multiple characters and events. With the advent of large datasets such as COCO [1, 19] and the image synthesis models [22, 23, 26, 28, 36, 37] pairing images with natural language descriptions became possible without enormous efforts in contrast to discriminative methods. The latter methods are carried out with excessive efforts, such as maintaining massive image resources and developing algorithms for retrieving, filtering, and selecting images [11]. Generating images that match the input text has wide applications in different domains, including news, communications, and business. One important application of pairing text with images is literacy development. In fact, children with learning difficulties can understand vocabulary and concepts better with images rather than raw text. In addition, new language learners can comprehend the vocabulary meaning when it is associated with representative images.

Recent works [17, 18, 26, 36] have shown that generative adversarial networks (GANs) [5] are immensely useful in generating realistic images. GAN-based approaches have demonstrated great success in generating high-resolution images with photorealistic details conditioned on textual descriptions [20, 24, 26, 36, 37], semantic segmentations [18] and scene graphs [14, 17]. However, successful results have been limited to generating images of single objects, often training one model per object class. Indeed, they still fail in generating images with recognizable objects when multiple objects are given. Due to the complexity of learning a direct text-to-pixel mapping from general images, as argued in [10], existing approaches still have problems in generating adequate images for complex text descriptions. Indeed, most works approach this task by generating images for short textual descriptions using GANs. However, because of this restriction, they use only object information in given text descriptions for rendering a specific object, leading to a poor layout of multiple objects in generated images [17] such as birds or flowers with simple backgrounds. Story text, conversely, is different from short textual descriptions, which emphasize multiple objects and semantic coherency more rather than single objects and simple descriptive text. Specifically, the task of story text visualization is to generate a sequence of images to narrate the story written in multiple sentence paragraphs. While most of the existing methods focus either on semantic compliance with the description on the image level [14] or on the scene layout level [27], few focus on both semantic and layout levels [4, 16, 32]. Notably, generating complicated real-world images such as COCO [19] remains a challenging task due to the following three challenges. First, in story text, the sentences are related to multiple objects with various semantical relationships. In this case, the presence of multiple objects with details of each object, and how to localize them all to reflect given relations becomes crucial for better image generation. Second, information about the appearance of the object in the story is neither explicitly available nor ready to integrate into the image generation, which makes it harder for any model to generate the desired object images. Third, in story visualization, the clarity of objects in terms of shape and how well the generated image matches the story text and the image quality are important details in realistic images. In addition, the consistency of the characters across all images is essential for promoting story understanding.

In this paper, we propose a new approach for generating images that represent the input text generatively. The approach is decomposed into three phases: story text understanding, object layout prediction, and image generation and refinement. Specifically, to address the first challenge, we focus on each sentence of the text and reduce its complexity to facilitate the information extraction process. We extract key phrases and transform them into a semantic representation. Moreover, for the second challenge, we follow the sg2im model [14] to generate the object layout for each character and mitigate the loss of object information by aggregating all individual object information (including basic geometric relations) from the scene graph. Finally, for the third challenge, we adopt the StackGAN for final image refinement under the guidance of the object layout module. In particular, our proposed framework provides additional information for the image generation process by incorporating an object layout module that captures information on the appearance of the image's objects.

The remainder of this paper is organized as follows. Section 2 scrutinizes related works of text-to-image generation approaches. Section 3 presents our proposed approach to generate images of multiple objects with specified relationships. Section 4 describes the settings for our experiments. Section 5 discusses the results and the evaluations, while Section 6 concludes the paper and highlights potential research directions.

2 Related works

In the last decade, significant GAN-based works have been carried out for generating representative text images [5, 12, 21, 28, 32] that are known as text-to-image synthesis methods. These methods are important for many applications, such as art generation and computer vision. Most GAN-based text-to-image generator models have demonstrated the flexibility to generate images that have been exploited in different tasks, such as image generation from textual descriptions [28, 36–38]. GANs can generate images from a noise dimension, i.e., a latent random variable [5], and have received attention because they produce sharper images compared to other generative models [21]. There are many other GAN variations; for instance, multistage GANs were developed to obtain high-resolution images, where each stage corrects the defects and adds details to the previous stage. Zhang et al. [38] proposed StackGAN to generate larger size images via a sketch-refinement process in two stages. Zhang et al. [37] proposed an improved version of StackGAN that uses multiple generators and discriminators in a tree-like structure. A later enhancement was proposed by Xu et al. [36] where they employed an attention mechanism that allows the network to focus on a single word from an input sentence or a specific region of the image at a time. Specifically, the attention mechanism works not only on a global sentence vector but also at the word level for fine-grained text-to-image generation. For instance, it focuses on object attributes when synthesizing different regions of the image.

Apart from generating images directly from noise with GANs, there has been significant work on conditioning the generator, discriminator or both on additional information, namely, conditional cGAN [22]. The conditioning problem in cGAN is modeled as a distribution of images given some prior information, for instance, providing object labels as an additional input to both the generator and discriminator. Conditional models have been studied extensively, and various approaches have been suggested to address it; however, many of them [12, 36, 38] focus on a single sentence to generate a single image. Other approaches suggested splitting the conditioning problem in different ways. An interesting approach was proposed by Mirza et al. [22], where the idea of conditioning both the generator and discriminator on extra information such as class labels was presented. Reed et al. [28] proposed using conditional

GAN with adversarial training of a generator and a discriminator to improve text-to-image generation. In this latter work, the authors were the first to propose a solution with promising results for the problem of image generation from text. They divided the problem into two main subproblems by finding a visually discriminative representation for the text descriptions and using this representation to generate realistic images.

Various approaches have been proposed to control the image generation process by conditioning on a class label [23], a text description [21, 28, 37], an auxiliary classifier [23], a semantic layout [10, 14] a dialog [30] or a semantic segmentation map [13]. More specifically, Zhang et al. [38] conditioned their generator and discriminator on caption encodings where their StackGAN model generates images in multiple stages; stage-I generates a coarse low-resolution (64×64) image, and stage-II generates the final high-resolution (256×256) image. This stacking of models resulted in generating highly photorealistic images at higher image resolution compared to previous works [3, 12]. However, they did not train their two stages in an end-to-end fashion. First, they trained stage-I until completion and then used the obtained output model from stage-I to train stage-II. The work in [37] builds StackGAN++ on top of StackGAN with a tree-like generator architecture for three or more generators and more stable training behavior. Based on StackGAN++, Xu et al. [36] increased the number of stages, trained them in an end-to-end fashion, added an attention mechanism over the captions, and added a novel attentional multimodal similarity model using the deep attentional multimodal similarity model (DAMSM) for text embedding. DAMSM learns two neural networks that map image subregions and sentence words to a common semantic space. This helps to measure the image-text similarity at the word level to compute a fine-grained loss for image generation, thus guiding the training loss and significantly increasing performance.

Story visualization, however, is different from short textual descriptions, which emphasize semantic coherency between story objects more than short descriptive text. A story text can contain different objects, scene backgrounds, events, locations, etc. To address such challenges, Lee et al. [16] proposed StoryGAN for story visualization that employs a context encoder to track the story flow and two discriminators at the story and image level to enhance the quality and consistency of generated images. However, due to the well-known difficulties of training generative models, such as instabilities in the training procedure [29], these works are limited to specific domains, such as cartoon characters [16]. In these specific domains, the image generation process is stable because the structures are much easier, and the quality of the generated image is usually not stable in most cases. Therefore, it is hard to directly apply generative models in complex scenarios such as scene generation for stories in the wild. Unlike GAN, Tan et al. [32] proposed text2scene as a sequence-to-sequence framework [31], which learns to sequentially generate objects and their attributes at every time step by attending to the words in the input text and the current status of the generated scene. However, their approach is restricted to composition tasks of abstract scenes.

Worth mentioning models attempted to resolve the semantic complexity of sentences to support semantic relevance enhancement. Specifically, semantic relevance enhancement focuses on improving the correlation between ground-truth text and the generated image. Notably, the author of AttnGAN [36] used StackGAN as the base model and proposed an attention mechanism by associating the subregions in the resulting image with the most relevant words in the input text. An interesting approach is the MirrorGAN [26], which incorporates a pretrained text redescription recurrent neural network (RNN) to better align the images with the given texts and to validate whether the generated images are consistent with the input texts. However, these GAN-based models do not specifically deal with the generation of multiple objects and their relationships that can occur in a story text. In addition, when they encounter a description language sentence that includes a plurality of objects and relationships of complex objects, they produce poor results. For instance, these models cannot identify such issues in sentences with more than one object or more than one instance, which results in poor image quality.

The idea of integrating object layout for image generation has also been explored by recent works. For instance, some works constructed a scene graph [10, 14], and others constructed a scene layout [17] or an object-driven semantic layout [18] by predicting bounding boxes and segmentation masks for all objects. The authors in Obj-GAN [18] used image captions to generate bounding boxes of specific objects within an image and predict the object's shape within each bounding box. Obj-GAN improved upon the work of [10] in many aspects. Specifically, for the task of box generation, the authors in Obj-GAN employed an attentive seq2seq model that captures the correspondence between an object label and a box. Their model showed better results than the seq2seq model used in [10]. A similar work [9] used an object path module to generate multiple objects at different positions using global and object pathways.

Our approach is different from the previously discussed works for several reasons. First, in a story text, the consistency of images emphasizes the continuity between consecutive image frames and supports story understanding. Second, in our case, we need to preserve the characters and their instances for the entire story. Third, to understand the story text, the relationships between the characters must be visualized in the generated images, which poses a major challenge for the recent GANs while we can properly address it and produce very good outcomes. Finally, in story text, the scene background may change within the same sentence; however, recent GANs cannot explicitly discriminate between the background, i.e., the background of scene image and foreground images, while our approach succeeds in solving this problem.

The main contribution of our approach is to use an object layout module, specifying the *subject relation object* from the scene graph, to localize the objects' bounding boxes while the relations among these objects are consistent. In addition, we stack and condition a StackGAN on the object layout to render realistic detailed objects that keep their relations even for complex text descriptions. Our approach is compared with other state-of-the-art approaches dealing with the same problem for image generation using the COCO dataset. The obtained results clearly show that our approach generates higher quality images that are more visible.

3 The approach

We follow a generative adversarial approach for scene generation and propose a two-stage architectural model as follows. First, in stage-I, we generate low-resolution images of the objects using the scene graph framework [14]. This phase is based only on averaged object label embeddings to represent objects in the scene layout. We then construct an object layout module to aggregate individual object labels and serve as image generation guidance for stacked image generation. Specifically, we build upon the StackGAN model introduced in [38] to generate high-resolution images by taking the 64×64 images generated from the sg2im framework [14] and the constructed object layout. Figure 1 provides an overview of the model architecture that we build as follows:

1. *Conversion of the story text into scene graphs*: We preprocess the COCO caption, i.e., story text into the form of triplets. Each triple is formed of a < *subject*, *relation*, *object* > semantically connected in the text.

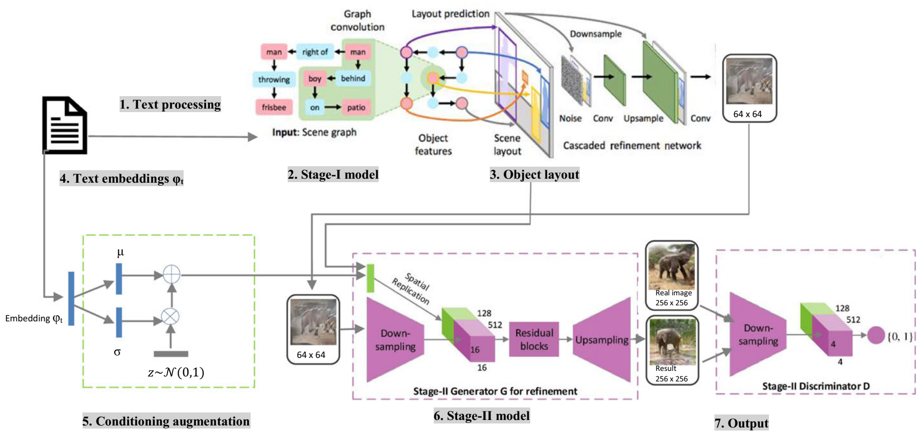


Fig. 1 The proposed framework overview (graphics partially adapted from StackGAN [38] and sg2im [14])

2. *Generation of low-resolution images*: We use triplets to generate images using GAN stage-I.
3. *Object layout construction*: We aggregate a predicted bounding box, a spatial binary mask and individual object feature for each object. Then, we concatenate them.
4. *Text embedding*: We embed the text as an input variable into a fixed-length vector of 1024 dimensions using a pretrained character level embedding encoder.
5. *Conditioning Augmentation (CA)*: We feed the embedding φ_t into a fully connected layer to generate means and standard deviations, which in turn produce the conditioning variable inputs for the generator in the next step.
6. *Image generation*: We apply a stage-II generator to generate a high-resolution image (256 × 256) with a stage-II discriminator. We feed the generator with conditional variable inputs concatenated with the low-resolution image and object layout tensor to generate the high-resolution image.
7. *Output image with computed K nearest neighbor (KNN) score*: We compute the KNN score for each output to assess the quality of every single image.

3.1 Story text processing

We process the story text as follows: Initially, key phrases are extracted using a part of speech tagger dealing only *with NN* for a noun and *V* for a verb. In fact, our objects are built upon 181 labels of the COCO-stuff dataset. We consider key phrases containing the geometric relationships *left of, right of, above, below, inside,* and *surrounding,* which gives the spatial relation between objects. Then, the extracted key phrases are manually translated from Arabic to English. Eventually, we extract all the triples <object, relationship, object> using the Stanford scene graph parser [33].

3.2 Stage-I image generation

For stage-I, we adopt the baseline approach for generating images from scene graphs proposed in [14]. The architecture consists of three main modules: a graph convolution network (GCN), a layout prediction network (LN) and a cascade refinement network (CRN). First, the GCN takes a scene graph as an input and computes new vectors for each node and edge in the graph. The GCN outputs an embedding label vector for each object. Then, these object embedding vectors are used by LN to compute a scene layout. This layout is computed by predicting a segmentation mask and bounding box for each object. Subsequently, given a scene layout, the CRN is responsible for generating an image that respects the object positions in the scene layout. The CRN consists of a series of convolutional refinement modules. Finally, an image discriminator network ensures that the overall appearance of generated images is realistic; it classifies a set of image patches as real or fake. The object discriminator network classifies each object as real or fake and ensures that each object is recognizable using an auxiliary classifier [23] that predicts the object's category.

However, the sg2im model lacks information about the appearance of the object, which makes it harder for the model to synthesize the object images for all object categories. This is because having the same scene graph represents many distinct images from the training dataset. In addition, the intermediate representation between the input graph domain and the output image domain loses any context from the scene graph, reducing the quality of the image generation stage. To overcome these drawbacks in our model, we add an object layout module to add the capability of reasoning out the appearance of the objects in the scene for stacking stage-II.

3.3 Object layout module

In the case of story text with multiple characters and events, it is necessary to automatically localize all the objects in the image so that they reflect, to some extent, the given relationships between them. This consideration is crucial for better image generation to support the story visualization task. Indeed, recent works [10, 14, 17, 18, 34] employ different techniques such as prediction networks to estimate a scene layout that gives initial and refined layout [34] or gives predictive values for the object appearances [10, 14, 17, 18]. However, in [14], the authors used object embeddings only to represent objects in their layout without using other details from the scene graph. Another limitation is that at the end of GCN, all occurrences of each object are merged with an average pooling operation such that individual object and individual relationship information are lost. Specifically, all embeddings that are associated with the same object are averaged together, resulting in new embeddings for that object. Such techniques may result in a failure to generate the correct object layout because they first predict the approximate average box and mask over the images corresponding to the given scene graph, and second, they do not consider the constraints specified by the scene graph.

To consider the required object label specified in the scene graph in the layout prediction, we introduce an object layout module to aggregate the features of all objects and their relations specified in the scene graph. In practice, we employ a fully connected network (FCN) to pool individual object information from the scene graph. In our case, the FCN is a multilayer perceptron (MLP) [25] that aggregates embeddings using 384 input nodes, 512 hidden nodes and 181 output nodes. MLP is known as a feedforward neural network and has the capability of arbitrary input-output mapping. In our case, FCN receives the embedded triplet and produces a score for each object label. Mathematically, an MLP comprising n_1 input nodes, n_2 hidden nodes, and n_3 output nodes is expressed as follows:

$$y_r = f_2\left(b_2 + \sum_{q=0}^{n_2} w_{qr}^2 * f_1\left(b_1 + \sum_{p=0}^{n_1} w_{pq}^1 * x_p\right)\right) \qquad r \in [1, n_3] \tag{1}$$

where:

$n_1 = 384$, $n_2 = 512$,$n_3 = 181$.
p, q, and r are the indices of the input, hidden, and output nodes, respectively
w_{pq}^1 and w_{qr}^2 are the weighting factors between layers
b_1 =0 and $b_2 = 0$ are bias weights to permit adjustments of the mean level at each stage
$f_1(g)$ is the ReLU [2] activity function of the hidden layer
$f_2(g)$ is the softmax [2] activity function of the output layer
y denotes the network outputs as a score for each object label

FCN outputs a score value for each predicted label. Thus, the obtained predicted label contains the averaged information from node embeddings as well as additional information from the scene graph. Thus, the newly obtained label is potentially more meaningful, remains true to its category class, and respects its dictated relations from the scene graph. An overview of the workflow of our method is shown in Fig. 2.

In practice, we obtain such object representation for each object as follows:

1. As a first step, we convert the nodes and edges of the scene graph into object and relation embeddings using a learned embedding layer. Additionally, a GCN is used to aggregate information across all objects and their relationships in the graph. The produced embeddings are used to form triplet embeddings, which are given as input to the FCN. As output, we obtain the predicted object labels.
2. We feed the averaged object embeddings into the layout prediction network to predict bounding boxes and the masks. The mask regression network consists of several transpose convolutions terminating in a sigmoid [2] nonlinearity so that elements of the mask lie in the range (0, 1); the box regression network is an MLP as a fully connected layer with linear ReLU activation. We obtain the bounding box defined by $\widehat{b}$ = (top, left, bottom, right) representing the top-left and bottom-right coordinates for each object, and a binary mask $\widehat{m}$ is given.
3. To construct the final representation of each object, we concatenate the predicted labels together with the bounding box coordinates and the binary mask.

Consider the example of *a man throwing a frisbee* (Fig. 2). To ensure that the objects of *man* and *frisbee* are preserved through the layout prediction, their embeddings must remain unchanged. To do that, we extract the embeddings from the scene graph namely, man_{emb} ,

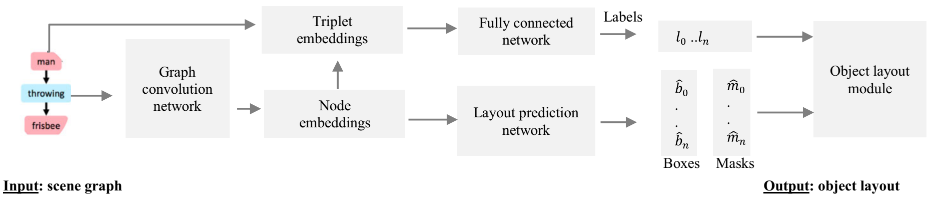


Fig. 2 Overview of the workflow in the object layout module

$throwing_{emb}$ and $frisbee_{emb}$. We do the same with averaged node embeddings from GCN, $man_{\overline{emb}}$ and $frisbee_{\overline{emb}}$. The resulting triplets shown below are fed into the FCN to add information from the scene graph as well as from the GCN.

$$man_{emb}, throwing_{emb}, frisbee_{\overline{emb}}$$

$$man_{\overline{emb}}, throwing_{emb}, frisbee_{emb}$$

In that manner, we maintain the identities of the objects and the relations between them according to the scene graph while also taking benefits from GCN embeddings, and the true predicted labels for the objects are 1 and 34 for *person* and *frisbee*, respectively. The predicted positions and sizes for each object are given in this form: top, left, bottom, right. These variables are normalized between 0 and 1 to be independent of the dimensions of the image.

During the training phase, the generated object layout requires learning the true labels, the desired positions, and the size for each object by using a ground-truth synthetic scene graph and bounding boxes from the training dataset. At test time, we use the predicted values from the prediction network and from the FC network. Finally, an embedded story text, the positions, sizes, and labels of the various objects within the scene graph, are input to the generator in stage-II in addition to a random noise vector.

3.4 Story text embedding

We encode the COCO caption, i.e., story text into the appropriate embedding φ_t to be prepared for input for stage-II. Therefore, we generate the text embedding φ_t with a 1024-dimensional vector using a pretrained character-based encoder, namely, Char-CNN-RNN [28], which gives very good results in text embedding.

3.5 Conditioning augmentation (CA)

The authors in [38] introduced the CA module to produce additional latent variable $\widehat{c}$ as input embeddings for the generator. Different from the static conditioning variable c, the variable $\widehat{c}$ is defined from an independent Gaussian distribution $\mathcal{N}(\mu(\varphi_t), \sum(\varphi_t))$ where the mean $\mu(\varphi_t)$ and diagonal covariance matrix $\Sigma(\varphi_t)$ are functions of the text embedding φ_t. As in variational autoencoders (VAE) [15], to ensure that the conditional space remains smooth and the model does not overfit, a regularization term enforces a standard normal distribution over the normal distributions of the text embeddings. The regularization term consists of the Kullback-Leibler (KL) divergence between the normal distribution of the embeddings and the standard normal distribution. The formula [38] is given below:

$$D_{KL}(\mathcal{N}(\mu(\varphi_t), \sum(\varphi_t)) \| \mathcal{N}(0, I_n)) \tag{2}$$

where I_n is the identity matrix and n is the dimension of the embedding φ_t

We use this formula to increase the enhancement and performance of the method as with StackGAN models [38]. The structure of CA is given in Fig. 1. The regularization idea from VAEs is used to perform the sampling. Hence, the network has the independence of learning the mean μ and the standard deviation σ of the embedding. For a text embedding φ_t, a fully connected layer with leaky ReLU activations computes μ and another computes σ and the sampled vector $\widehat{c}$ is obtained as shown in the following equation:

$$\widehat{c} = \mu + \sigma \odot \varepsilon \tag{3}$$

where $\varepsilon \sim \mathcal{N}(0, I_n)$ and $\odot$ is elementwise multiplication

We then concatenate the story text embeddings and object layout prediction matrices, and this is passed as input to the stage-II model. We assume, by conditioning image generation on explicit object layout predictions from stage-I, our method can generate images that are semantically meaningful and well-aligned with input text, for instance, by measuring the semantic relevance between the image and the text. We conduct extensive quantitative and qualitative experiments and evaluations on the MS-COCO dataset and demonstrate substantial improvement in generation quality over existing works in the domain of narrative stories.

3.6 Image quality enhancement

We focus on improving the quality of the images generated by our stage-I model since the images are commonly blurred, with only key layout and color that correspond approximately to the text description. Moreover, various details are omitted in these images since capturing fine-grained information from high-level text and fitting image data distribution simultaneously is intractable for a single GAN model. To address these problems, we adopt a conditional GAN topped upon the scene graph framework in stage-II, which refines the output for both high-resolution (i.e., 256 × 256) and realistic image generation. Specifically, for the conditioning, we use the generated object layout concatenated by text embeddings, with the same conditioning augmentation applied. The generator G obtains as input a randomly sampled noise vector z, the location and size of the individual bounding boxes, a label for each of the bounding boxes encoded as a one-hot vector, and an image-caption embedding. Similar to most generative models [38], we use the same minmax objective function as a loss function to train both models G and D simultaneously, where G and D explicitly learn a maximization, i.e., a minimization of the expected log-likelihood of a data distribution by alternating between maximizing the loss function $\mathcal{L}_D$ and minimizing the loss function $\mathcal{L}_G$:

$$\mathcal{L}_D = E_{(I,c)\sim p_{data}}[\log D(I,c)] + E_{s_0\sim p_{G0}, c\sim p_{data}}[\log(1-D(G(s_0,\hat{c}),c))] \tag{4}$$

$$\mathcal{L}_G = E_{s_0\sim p_{G0}, c\sim p_{data}}[\log(1-D(G(s_0,\hat{c}),c))] + \lambda D_{KL}(\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t)) \| \mathcal{N}(0,1)) \tag{5}$$

where:

$E_{(I,c)\sim p_{data}}$ is the expected value over all training data distributions p_{data}.
I is the real image
c is the conditional information for this image (e.g., an image caption φ, an object category label)
$\widehat{c}$ is the Gaussian latent variable concatenated with the encoded conditional information
$D(I, c)$ is the discriminator's probability estimation that real data pairs (I, c) are real
s_0 is the image generated from stage-I model p_{G0}
$E_{s_0\sim p_{G0}, c\sim p_{data}}$ is the expected value over all generated images $G(s_0, \widehat{c})$
$G(s_0, \hat{c})$ is the generator's output image
$D(G(s_0, \hat{c}), c)$ is the discriminator's probability estimation that a fake generated image is real

$D_{KL}(\mathcal{N}(\mu(\varphi_t), \sum(\varphi_t)) \| \mathcal{N}(0, 1))$ is the KL divergence between the normal distribution of the text embeddings and the standard normal distribution

λ is a regularization parameter (1 by default)

3.7 Image quality assessment

Recently, many quantitative metrics have been developed to assess the performance of a GAN model, such as the inception score [7], FID [8], and GIQA [6]. We adopt the latter since it can separately assess the quality and diversity of generated images. Specifically, it employs the mean quality score to indicate the performance of the generative model. Assume the generative model is G, and the generated samples are $\mathcal{I}_g^i, i = 1, 2, \ldots, N_g$. Therefore, the quality score of generator G is calculated with the mean quality of N_g generated samples:

$$QS(G) = \frac{1}{N} \sum_{i}^{N_g} S\left(\mathcal{I}_g^i\right) \tag{6}$$

The GIQA model methods aim to solve the quality estimation from a probability distribution perspective. The model directly predicts the probability distribution of the real data; then, it can estimate the quality of a generated image by the estimated probability from the model. The model proposes two density estimation methods: the Gaussian mixture model (GMM) and the KNN model. We adopt the latter model to assess the quality of generated images since it performs well when the real data distribution becomes complicated, similar to our case, as argued in [6], where the Euclidean distance between generated images and nearby real images in feature space could also represent the probability of generated image. Accordingly, the image quality score of any generated image is computed as follows:

$$QS\left(\mathcal{I}_g\right) = \frac{1}{N} \sum_{k=1}^{K} \frac{1}{\|x - x^k\|^2} \tag{7}$$

where x is the feature of a generated sample and x^k is its k-th nearest real sample's feature.

4 Experiments

We conducted extensive experiments to evaluate our approach. We compared it with state-of-the-art approaches for image synthesis, namely, sg2im, StackGAN and AttnGAN, and we showed its performance in aspects of semantic matching, object recognition, realism, and image quality.

4.1 Datasets

We performed the experiments on COCO datasets [1, 19] with additional stuff categories, as shown in Table 1 below. We noted that the latter has 40 K images for training and 5 K images for evaluation with annotation including bounding boxes and segmentation masks. This

Table 1 Dataset details

Dataset	Train	Validation	Test
COCO 2014 [1]	83 K	41 K	41 K
COCO 2017 [19]	118 K (only 40 K are stuff annotated)	5 K	41 K
	25 K (effectively used by sg2im [14])	1 K (effectively used by sg2im [14])	2 K (effectively used by sg2im [14])

dataset contains 80 "thing" categories (people, cars, etc.) and 91 "stuff" categories (sky, grass, etc.). However, effectively, sg2im [14] used 25 K for training, 1 K for evaluation, and 2 K for testing, whereas StackGAN [38], AttnGAN [36] and our model used 83 K for training and 41 K for validation.

For computing the GIQA score for the generated images, we used the COCO 2017 [19] dataset for feature extraction.

We followed the procedure described in [14] to construct synthetic scene graphs from these annotations based on the 2D image coordinates of the objects using six mutually exclusive geometric relationships: *left of, right of, above, below, inside,* and *surrounding*. We also ignored objects covering less than 2% of the image and used images with 3 to 8 objects. For general training, we applied the same procedure used in the GAN architecture that was modified with our proposed approach based on sg2im [14] and StackGAN [38], as shown in Fig. 1.

For the story text dataset, we prepared 20 stories for children from the animal domain that have a great potential impact on engaging children in the learning process and keeping them highly motivated. The selected stories have 80 key phrases with simple narrative structures in introducing concepts using animal characters and their common behaviors, such as running, eating, jumping, and hunting. The sentences are relatively short with up to six nonfiction terms.

Example:

"*Two elephants are in a grassy field. They are close to two sheep. They are eating grass.*"

The characters, objects, location, and background were explicitly mentioned in the text and realistic as shown in Table 2 below.

Table 2 An excerpt from stories' details

Characters (subject, object)	Keywords	Spatial relations	Background
Elephant	Elephant walking Elephant eating Elephant running Elephant drinking Elephant standing	left of, right of, above, below, inside, surrounding	grassy field
Sheep	Sheep walking Sheep eating Sheep running Sheep drinking Sheep standing	left of, right of, above, below, inside, surrounding	grassy field

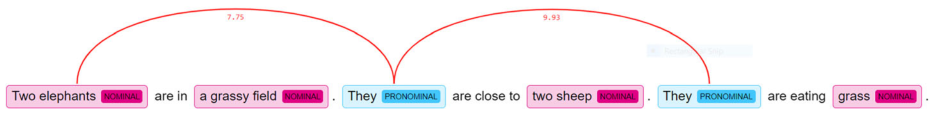


Fig. 3 Coreference resolution applied to the sentences in the previous example using NeuralCoref

All the sentences were preprocessed using NeuralCoref[1] to reduce the semantic complexity of the story text. For instance, the ambiguous pronouns in the following sentences were adjusted as depicted in the following graph shown in Fig. 3.

After the coreference resolution and manual adjustment, we obtain the following final representation for the sentences:

> *"Two elephants are in a grassy field. Two elephants are close to two sheep. Two sheep are eating grass."*

4.2 Training details

We trained the two-stage model separately but with the following hyperparameter settings: we used the Adam optimizer with a learning rate of 10^{-4} for all trainings and a batch size of 24 for 100 epochs. However, when attempting to use a larger batch size, it resulted in a mode collapse. The text embedding dimension parameter presented in stage-II was of size 1024, while all other parameters were identical to baseline models [14, 38]. Here, we added the object layout matrix during the training phase concatenated with the text embedding and object labels as input for the second generator following a similar procedure proposed in [9, 18]. We also followed the same training given in [14]. For better visualization, our stage-II additional information was provided with up to three bounding boxes and aligned object category labels per image. Each line of the story text was considered as an image caption as well as additional information. Detailed qualitative and quantitative analysis of our results and comparisons against the baseline models are discussed in the next sections.

5 Evaluation

We compared our approach with three state-of-the-art methods: sg2im [14], StackGAN [38], and AttnGAN [36]. We followed their dataset preprocessing and conducted both quantitative and qualitative assessments as follows.

5.1 Quantitative results

We demonstrate the image quality improvement of our model by assessing the image quality of the three baseline models against our model, as shown in Figs. 4 and 5. There are several assessment algorithms, such as the inception score [7] and FID [8], to evaluate the performance of a generative model in a score-based manner with regard to two aspects: realism and diversity. For our approach, we adopted the GIQA [6] method because it can separately assess the realism and diversity of the generated images contrary to previous methods, the inception

[1] github.com/huggingface/neuralcoref

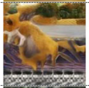
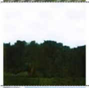
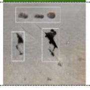
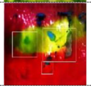
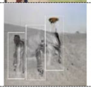
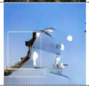
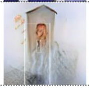
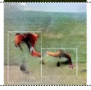
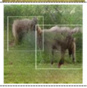
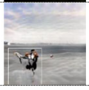
Caption	A person above a playingfield and left of another person left of grass, with a car left of a car above the grass.	One broccoli left of another, which is inside vegetables and has a carrot below it.	Three people with the first two inside a fence and the first left of the third.	A person above the trees inside the sky, with a skateboard surrounded by sky.	A tie above clothes and inside a person, with a wall panel surrounding the person.	A tree right of a person left of a horse above grass, with clouds above the grass.	An elephant above grass and inside trees surrounding another elephant.	Clouds above a boat and a building above a river, with trees left of the river.
Sg2im [16]								
StackGAN [23]								
AttnGAN [9]								
Ours								

Fig. 4 Sample images generated from the baseline models and our model using captions from coco dataset

score [7] and FID [8], where these two aspects are assessed and summed. Hence, we assess the quality of the generated images, and specifically, we use the KNN method with K = 1000 and rank the quality of these samples based on it.

Note that a lower KNN score denotes better KNN-GIQA results. This score was normalized to [0,1] for comparison with other scoring methods. First, we extracted the features from all images in the COCO 2017 dataset, and then we computed the KNN score based on these features, as shown in Figs. 4 and 5. For KNN score computation, all images were downsampled to the same image size of 299 × 299. The downsampling process resized and rescaled images where Gaussian smoothing was performed to avoid any aliasing artifacts.

5.2 Qualitative results

We show the results of each baseline, and we restrict the test samples to a maximum of three objects in the image generation for better visual perception. The model sg2im uses a scene

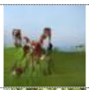
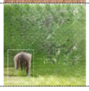
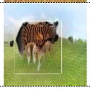
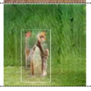
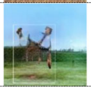
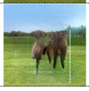
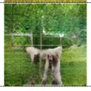
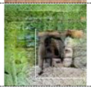
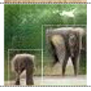
Caption	A baby elephant is walking through the jungle.	A zebra is standing on a field.	A giraffe is walking next to a fence.	Two giraffes are standing in a field.	A horse is standing in a field.	A brown cow under a tree in a grassy area.	A bear walking in the woods next to some trees.	An elephant standing next to a smaller elephant.
Sg2im [16]								
StackGAN [23]								
AttnGAN [9]								
Ours								

Fig. 5 Sample images generated from the baseline models and our model using simple captions adapted from COCO dataset

graph framework to generate a single image for each scene graph triple. The model StackGAN generates a single image for each sentence. We also show the images generated from our two-stage model with and without the object layout module. For all models, we generated one image for each key-phrase test-set and downsampled their output to 64 × 64 for fair and proper comparison with our approach.

As shown in Figs. 4 and 5, the first baseline sg2im [14] captured the semantic context provided by the graph and generated scenes with multiple objects and even multiple instances of the same object type. However, (i) it failed to locate the objects in the right positions because the graph convolution has benefits beyond simply predicting object positions, which differed in most cases with the desired real object positions; and (ii) the generated images were of poor quality. The baselines StackGAN [38] and AttnGAN [36], produced visually pleasing and high-resolution images but completely failed to generate recognizable objects or shapes and therefore failed to capture any semantic context provided from the scene graph framework. In addition, the images had poor color contrast, and no sharp borders were available.

Our model, however, illustrates that we can control image generation through the injection of object layout features during the generation process. This enabled us to control the object category, location, and the size of multiple objects within a given image. The generated bounding boxes and object labels thereby facilitated the generation of more complex scenes. We preserved the location of objects and sizes while adding new objects to the image created in the previous step in accordance with the relationships defined in the scene graph. This preservation of object location was a missing feature in both baseline models because they generated the objects at their positions in an end-to-end manner. Regarding the addition of objects, we restricted the object to three entities due to the size restriction of the scene, since adding more objects makes the scene cluttered and visually unclear.

To evaluate whether our method generates objects at the given positions, we generated images that contain multiple objects and inspected them visually. Figure 4 shows some example images, where we visualized the results of baseline models and our method using the same caption text. The corresponding KNN scores for each output are averaged and presented in Table 3. Noticeably, our method obtained the best KNN scores in most of the outputs by showing a total score of 1.53 compared to 1.69, 1.63 and 1.71 for the sg2im, StackGAN and AttnGAN models, respectively, which was a considerable improvement in image quality, as seen from the KNN averaged score values shown in Table 3 below.

5.3 Ablation study

To evaluate the performance of our method using the object layout module, we compared the image quality of two ablated versions of our model, as shown in Fig. 6. The results of the evaluation demonstrated the importance of this module in our model. To do that, we selected

Table 3 Averaged and normalized KNN-GIQA score of all compared methods

Dataset	Resolution	Method	Averaged KNN score	Normalized KNN score
COCO 2017 [19]	64×64	Sg2im	1.69	0.29
COCO 2014 [1]	256×256	StackGAN	1.63	0.24
COCO 2014 [1]	256×256	AttnGAN	1.71	0.55
COCO 2014 [1]	256×256	Ours	**1.53**	**0.15**

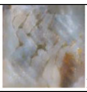
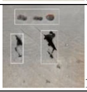
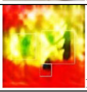
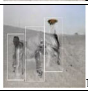
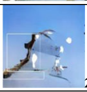
Caption	A person above a playingfield and left of another person left of grass, with a car left of a car above the grass.	One broccoli left of another, which is inside vegetables and has a carrot below it.	Three people with the first two inside a fence and the first left of the third.	A person above the trees inside the sky, with a skateboard surrounded by sky.
Ours without object layout	1.48	 1.86	2.44	2.37
Ours with object layout	 **1.31**	 **1.67**	 **1.40**	 **2.15**

Fig. 6 Sample images generated from the ablation study of our model including KNN scores

four captions from Fig. 4 and generated the images following two versions, namely, with or without the object module. We clearly notice that when we add the object layout module to stage-II, the generated images show recognizable objects compared to the version without the object layout module. We also observed a relatively better KNN score for our model compared to the version without the object layout module.

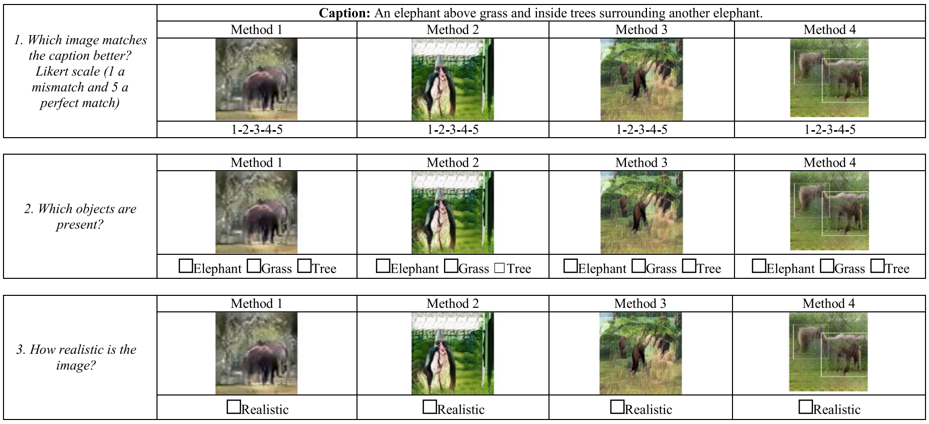
5.4 User study

Due to the subjectivity of inspecting the generated images, we further conducted a human evaluation on the generated images in terms of semantic matching (i.e., how accurate the generated image fits with the given input), object recognition (i.e., which objects were recognizable in the generated image) and realism (i.e., how realistic or natural the generated image looks). We compare our method with sg2im [14], StackGAN [38] and AttnGAN [36], three state-of-the-art methods used for generating images from object labels and image captions. Though the three models use different input modalities and different architectures, it is expected that they should generate similar images, since they share the same COCO dataset in terms of object categories and correspondence between the captions and objects.

We followed the work of [14] for user evaluation and provided users with a questionnaire with 20 sections, where each section contained a caption and four generated images using the three baseline models and our model. All four methods were shown anonymously and in a random order to the user where each image is accompanied by a check box for one user group and by a Likert scale for another user group. In addition, a list of objects was also presented for each generated image, and the users were asked to tick the objects that appeared in each of the images. Finally, the user chose whether each generated image appeared realistic. Note that we downsampled all generated images to 64×64 to compensate for differing resolutions. In practice, we used antialiasing as a high-quality downsampling filter in image processing. Figure 7 shows the excerpt of our questionnaire from the user study.

We involved two different user groups in our evaluation, successively. The first group (group A) has twenty-one volunteer participants, from our institution and have different scientific backgrounds in computing; most of them were postdoctoral fellow and research assistants in the Department of Computer Science and Engineering at Qatar University. Eleven participants had strong image processing backgrounds, while the other ten had computer science and artificial intelligence backgrounds. On the other hand, the second group (group B) has thirty volunteer participants; all of them were students from mechanical and civil Engineering Departments at Qatar University.

We tested twenty input sentences resulting in sixty-eight and eighty generated images by the four methods, for group A and group B respectively. We asked group A to evaluate the



	Caption: An elephant above grass and inside trees surrounding another elephant.			
1. Which image matches the caption better? Likert scale (1 a mismatch and 5 a perfect match)	Method 1	Method 2	Method 3	Method 4
	1-2-3-4-5	1-2-3-4-5	1-2-3-4-5	1-2-3-4-5

	Method 1	Method 2	Method 3	Method 4
2. Which objects are present?	☐Elephant ☐Grass ☐Tree	☐Elephant ☐Grass ☐Tree	☐Elephant ☐Grass ☐Tree	☐Elephant ☐Grass ☐Tree

	Method 1	Method 2	Method 3	Method 4
3. How realistic is the image?	☐Realistic	☐Realistic	☐Realistic	☐Realistic

Fig. 7 An excerpt of our questionnaire used in our evaluation study

generated images using a check box only, while group B was asked to evaluate the generated images using the Likert scale of 1 to 5, where 5 means the best match and 1 means the worst. The user evaluation was conducted to assess the image quality i.e., text-to-image semantic matching, object recognition and image realism. We further analyze the outcomes of the user evaluation and summarize the results from the two groups in Tables 4 and 5. The obtained results show that our method significantly outperformed the baseline methods in two aspects. First, in terms of the semantic matching between a caption and an image, the summarized results from Table 4 show that our model achieved an accuracy of 48.6% compared to 34.4%, 12.9% and 11.2% for that of the sg2im, StackGAN and AttnGAN models, respectively. Most of the users (58.6%) preferred the images generated by our approach, showing that our method gives more semantically matching caption image pairs. Likewise, the summarized results from Table 5, show that our method reached a mean value of 2.55 compared to 2.33, 2.03, and 2.29 that of the sg2im, StackGAN and AttnGAN models, respectively. Moreover, Fig. 8 shows that our method ranked first in getting 5 as the best result within all 20 images evaluated by group B. This is mainly attributed to the injection of the caption in our model in two different input modalities. Notably, the semantic matching between the caption image pairs is supported in two different ways: first, the caption is translated into a scene graph in stage-I, and then it is used as a caption embedding in stage-II.

Second, in terms of realism, eleven users from group A found that our generated images looked more real than the images generated by the three other baseline models, giving a

Table 4 Semantic matching, object recognition and realism percentages for the baseline models and our method (results from group A)

Method	Semantic matching	Object recognition	Realism	
			Yes	No
Sg2im	34.4%	41.9%	38.2%	61.6%
StackGAN	12.9%	32.3%	24.6%	75.3%
AttnGAN	11.2%	28.1%	30.9%	75.4%
Ours	48.6%	38.1%	47.5%	52.4%

Table 5 Semantic matching, object recognition and realism percentages for the baseline models and our method (results from group B)

Method	Semantic matching		Object recognition	Realism	
	Average	percentage		Yes	No
Sg2im	2.33	46.6%	54.6%	41.5%	58.5%
StackGAN	2.03	40.5%	48.5%	32.6%	67.3%
AttnGAN	2.29	45.8%	52.2%	44.6%	55.5%
Ours	2.55	50.9%	54.0%	45.5%	54.5%

percentage of 47.5% compared to 38.2%, 24.6% and 30.9% for the sg2im, StackGAN and AttnGAN models, respectively. Likewise, fourteen users from group B found that our method produces more real images, as can be seen from Table 5 showing the highest percentage of 45.5% among others. The reason behind this feature is that the added object layout module in stage-II of our model explicitly adds the true object labels to the generation step, resulting in more recognizable objects to some extent. This helps in generating more real images, in addition to the availability of the discriminator resulting in the generation of highly photorealistic images, as is known in all GAN-based architectures. Our model generated realistic images for almost half of the captions, as shown in Tables 4 and 5.

The two abovementioned aspects, namely, semantic matching and realism, correlate together, as we can see from the results that models with higher semantic matching accuracy produce more realistic images than other models with lower semantic matching accuracy.

For the object recognition aspect, we measured the fraction of objects that users can recognize in images from each method. The baseline model sg2im produced more recognizable objects since it was trained on object categories from the COCO 2017 [19] dataset than the other methods. In addition, the input modality in the sg2im method allowed objects to be used that were not necessarily mentioned in the caption. Although sg2im showed 41.9% and 54.6% recognizable objects according to group A and group B, respectively, it did not produce recognizable objects in many cases, since the result showed that most of the objects were not recognizable. Our model showed a fair result in recognizing objects compared to the three other baseline models.

In summary, according to the objective evaluation though GIQA-score and the subjective evaluation through user evaluation, our method outperformed baseline models in generating

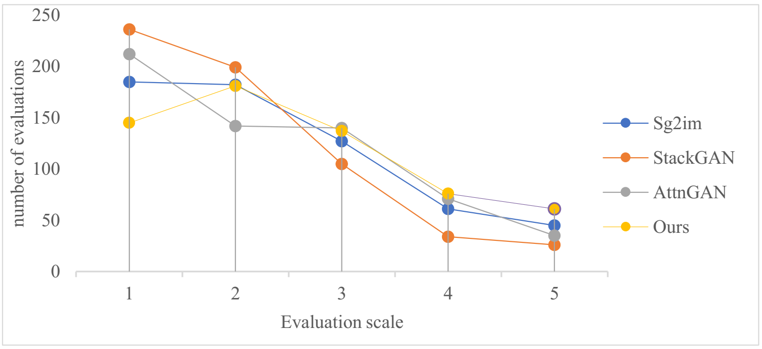


Fig. 8 Semantic matching of the user assessment for all methods (results from group B)

images with higher quality, accurate semantic matching between text and image, and realistic images. As a result, our method can generate images that are clearer and more visually pleasing, semantically meaningful, and well-aligned with the captions. In addition, our method provides a fair result for recognizing objects, which we believe enhances its ability in using other techniques, such as an object shape predictor, as given in [10]. Indeed, the results obtained in our experiments are very promising, as the image generation module learns how to produce images with respect to textual descriptions, object positions, and object categories while achieving the required image quality with high resolution. The results show that as the text become more complex, our method is increasingly able to turn the texts into realistic images when compared with the other models.

6 Conclusion

In this paper, we addressed the challenges of visualizing a story text that contains multiple characters and representative semantic relationships. We proposed a novel approach for generating images using a two-stage model architecture inspired by two state-of-the-art GAN-based models for image generation. We employed the object layout module as a technique to guide the image generation process towards producing more meaningful and realistic images. Through extensive evaluation and qualitative results, we demonstrated that our approach can indeed generate an image of very good quality representing the main objects in the text. The object layout module can control the location, size and object category while finalizing the fine-grained image generation in stage-II. Our work can be applied in different domains specifically in the education of children with images rather than using raw texts. Indeed, many studies demonstrate that children are more likely to learn concepts from realistic images than from fictive stories [35]. While our proposed approach produces higher quality images than the existing baseline approaches, the object shape, the location, and the size attributes sampled are still, in some cases, not adequate with a realistic image and need further improvement. In these cases, a manual adaptation may be applied to rectify the overall layout of the proposed image and make it more realistic. Indeed, we believe that using ground-truth data such as real layouts can help in this direction in generating natural images and limiting or eliminating manual intervention. Specifically, the use of auxiliary loss for the discriminator, e.g., by addressing the object recognition task, can also lead to some improvement~~s,~~ as has been observed in prior work on other image generation tasks.

An interesting research direction we aim to explore further is to enrich the underlying scene graph modality to capture the story text semantics as well as to expand the vocabulary size to other specific domains rather than the COCO dataset general domain. Moreover, due to the semantic complexity of the text and the limited quantity of image-caption paired data in the COCO dataset, the model might not always be able to understand which objects in the image correspond to which word in the caption (e.g., exact matching). In many cases, the model is yet able to generate images that exhibit strong global consistency but do not produce recognizable objects. Our approach would be enhanced by additionally predicting the specific object shape within each bounding box as done for other approaches. To this effect, we further plan to explore other datasets to resolve this issue limitations or build our own dataset (i.e., a bimodal large-scale corpus).

We believe that more fine-grained control over the image layout can also lead to better image quality. In the future, we plan to generate scene graphs automatically from the text input. We plan to investigate whether the quality of images can be improved by using an attention mechanism during generation. In particular, we plan to use MirrorGAN to better align the images with the given texts. This can also potentially make the task of only generating certain regions in the image easier. Furthermore, we plan to explore better architectures for image generation through layouts for higher image quality.

Acknowledgments This work was made possible by NPRP grant #10-0205-170346 from the Qatar National Research Fund (a member of Qatar Foundation). The statements made herein are solely the responsibility of the authors.

Funding Open access funding provided by the Qatar National Library.

References

1. Caesar H, Uijlings J, Ferrari V (2018) Coco-stuff: Thing and Stuff Classes in Context, *ArXiv:1612.03716*
2. Chigozie N, Ijomah W, Gachagan A, Marshall S (2018) Activation Functions: Comparison of trends in Practice and Research for Deep Learning, ArXiv abs/1811.03378
3. Denton E, Chintala S, Szlam A, Fergus R (2015) Deep generative image models using a Laplacian pyramid of adversarial networks, ArXiv e-prints
4. Gangyan Z, Zhaohui L, Yuan Z (2019) PororoGAN: an improved story visualization model on Pororo-SV dataset, in 3rd International Conference on Computer Science and Artificial Intelligence, Normal IL USA
5. Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets, In Advances in Neural Information Processing Systems, p. pages 2672–2680
6. Gu S, Bao J, Chen D, Wen F (2020) GIQA: Generated Image Quality Assessment, ArXiv:2003.08932
7. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A (2017) Improved training of wasserstein GANs, in Proceedings of the 31st International Conference on Neural Information Processing Systems, pages 5769–5779
8. Heusel M, Ramsauer H, Unterthiner T, Nessler B, Hochreiter S (2017) GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, in Advances in neural information processing systems. pages 6626–6637
9. Hinz T, Heinrich S, Wermter S (2019) Generating multiple objects at spatially distinct locations, in International Conference on Learning Representations, ICLR
10. Hong S, Yang D, Choi J, Lee H (2018) Inferring semantic layout for hierarchical text-to-image synthesis, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 7986–7994
11. Huang C-J, Li C-T, Shan M-K (2013) VizStory: visualization of digital narrative for fairy Tales, in Conference on Technologies and Applications of Artificial Intelligence, pages 67–72, USA
12. Huang X, Li Y, Poursaeed O, Hopcroft J, Belongie S (2016) Stacked generative adversarial networks, CVPR, vol. arXiv:1612.04357
13. Isola P, Zhu J-Y, Zhou T, Efros AA (2017) Image-to-image translation with conditional adversarial network, ArXiv:1611.07004
14. Johnson J, Gupta A, Fei-Fei L (2018) Image Generation from Scene Graphs, in IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 1219–1228, doi: 10.1109/CVPR.2018.00133, Salt Lake City, UT, USA
15. Kingma DP, Welling M (2013) Auto-encoding variational bayes, ArXiv, p. arXiv:1312.6114

16. Li Y, Gan Z, Shen Y, Liu J, Cheng Y, Wu Y, Carin L, Carlson D, Gao J (2019) StoryGAN: A Sequential Conditional GAN for Story Visualization, Conference on Computer Vision and Pattern Recognition (CVPR), vol. abs/1812.02784, pp. 6322–6331
17. Li Y, Ma T, Bai Y, Duan N, Wei S, Wang X (2019) PasteGAN: a semi-parametric method to generate image from scene graph. Adv Neural Inf Proces Syst 32:3948–3958
18. Li W, Zhang P, Zhang L, Huang Q, He X, Lyu S, Gao J (2019) Object-driven Text-to-Image Synthesis via Adversarial, CVPR, p. arXiv:1902.10740
19. Lin T-Y, Maire M, Belongie S, Hays J, Perona P, Ramanan D, Doll'ar P, Zitnick CL (2014) Microsoft COCO: Common Objects in Context, in European conference on computer vision, pages 740–755
20. Lu Y, Wu S, Tai Y-W, Tang C-K (2018) Image generation from sketch constraint using contextual gan, *Computer Vision ECCV* 2018. Lecture notes in computer science, vol 11220, pp https://doiorg/101007/978-3
21. Mansimov E, Parisotto E, Lei Ba J, Salakhutdinov R (2016) Generating Images from Captions with Attention, *ArXiv:1511.02793*
22. Mirza M, Osindero S (2014) Conditional Generative Adversarial Nets, *ArXiv:1411.1784*
23. Odena A, Olah C, Shlens J (2017) Conditional Image Synthesis With Auxiliary Classifier GANs, *ArXiv: 1610.09585*
24. Ouyang X, Zhang X, Ma D, Agam G (2018) Generating image sequence from description with lstm conditional Gan, ArXiv*:* 1806.03027
25. Popescu M-C, Balas VE, Perescu-Popescu L, Mastorakis N (2009) Multilayer perceptron and neural networks. World Scientific and Engineering Academy and Society (WSEAS) 8(7):579–588
26. Qiao T, Zhang J, Xu D, Tao D (2019) MirrorGAN: learning text-to-image generation by Redescription, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 1505–1514
27. Qiao X, Zheng Q, Cao Y, Lau RWH (2019) Tell me where I am: object-level scene context prediction, in Conference on Computer Vision and Pattern Recognition (CVPR), pages 2628–2636, Long Beach, CA, USA
28. Reed S, Akata Z, Yan X, Logeswaran L, Schiele B, Lee H (2016) Generative adversarial text to image synthesis, in Proceedings of the 33rd International Conference on International Conference on Machine Learning, pages 1060–1069
29. Salimans T, Goodfellow I, Zaremba W, Cheung V, Radford A, Chen X (2016) Improved techniques for training gans, Adv neural inf process syst*,* p. 2234–2242
30. Sharma S, Suhubdy D, Michalski V, Kahou SE, Bengio Y (2018) ChatPainter: Improving Text to Image Generation using Dialogue, ArXiv:1802.08216
31. Sutskever I, Vinyals O, Le QV (2014) Sequence to sequence learning with neural networks, In Advances in Neural Information Processing Systems(NeurIPS)
32. Tan F, Feng S, Ordonez V (2019) Text2Scene: generating compositional scenes from textual descriptions, in IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
33. "The Stanford Natural Language Processing Group," [Online]. Available: https://nlp.stanford.edu/software/scenegraph-parser.shtml. [Accessed 1 October 2019].
34. Vo DM, Sugimoto A (2020) Visual-relation conscious image generation from Structured-Text, in arXiv preprint arXiv:1908.01741
35. Weisberg DS, Hopkins EJ (2020) Preschoolers' extension and export of information from realistic and fantastical stories, Infant and Child Development, p. doi:https://doi.org/10.1002/icd.2182
36. Xu T, Zhang P, Huang Q, Zhang H, Gan Z, Huang X, He X (2018) Attngan: Fine-grained text to image generation with attentional generative adversarial networks, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1316–1324, doi: https://doi.org/10.1109/CVPR.2018.00143., Salt Lake City, UT
37. Zhang H, Xu T, Li H, Zhang S, Wang X, Huang X, Metaxas D (2017) StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks, *ArXiv:1710.10916*
38. Zhang H, Xu T, Li H, Zhang S, Wang X, Huang X, Metaxas D (2017) StackGAN: Text to Photo-Realistic Image Synthesis with Stacked Generative Adversarial Networks, in IEEE International Conference on Computer Vision (ICCV), pp. 5908–5916, doi: 10.1109/ICCV.2017.629, Venice, Italy