JMLR: Workshop and Conference Proceedings 101:284–299, 2019 ACML 2019

Realistic Image Generation using Region-phrase Attention

Wanming Huang WANMING.HUANG@STUDENT.UTS.EDU.AU
Faculty of Engineering and IT
University of Technology Sydney
81 Broadway, Sydney, Australia

Richard Yi Da Xu YIDA.XU@UTS.EDU.AU
Faculty of Engineering and IT
University of Technology Sydney
81 Broadway, Sydney, Australia

Ian Oppermann IANOPPER@OUTLOOK.COM
NSW Data Analytics Centre
2-24 Rawson Pl, Sydney, Australia

Editors: Wee Sun Lee and Taiji Suzuki

Abstract

The Generative Adversarial Network (GAN) has achieved remarkable progress in generating synthetic images from text, especially since the use of the attention mechanism. The current state-of-the-art algorithm applies attentions between individual regular-grid regions of an image and words of a sentence. These approaches are sufficient to generate images that contain a single object in its foreground. However, natural languages often involve complex foreground objects and the background may also constitute a variable portion of the generated image. In this case, the regular-grid region based image attention weights may not necessarily concentrate on the intended foreground region(s), which in turn, results in an unnatural looking image. Additionally, individual words such as "a", "blue" and "shirt" do not necessarily provide a full visual context unless they are applied together. For this reason, in our paper, we proposed a novel method in which we introduced an additional set of natural attentions between object-grid regions and word phrases. The object-grid region is defined by a set of auxiliary bounding boxes. They serve as superior location indicators to where the alignment and attention should be drawn with the word phrases. We perform experiments on the Microsoft Common Objects in Context (MSCOCO) dataset and prove that our proposed approach is capable of generating more realistic images compared with the current state-of-the-art algorithms.

Keywords: Image generation, Natural language processing, Generative adversarial networks

1. Introduction

Generating images from text descriptions is a challenging problem that has attracted much interest in recent years. Algorithms based on the Generative Adversarial Network (GAN) (Goodfellow et al. (2014)), specifically Deep Convolutional GAN (DCGAN) (Radford et al. (2015)) and conditional GAN (Mirza and Osindero (2014)), have achieved remarkable progress on various datasets. Works from Reed et al. (2016c) and Xu et al. (2017) have shown promising results in synthesizing images that contain a single object, such as on the

CUB (Welinder et al. (2010)) and Oxford-102 (Nilsback and Zisserman (2008)) datasets. However, synthesizing an image that models human poses or involves multi-object interactions usually lacks sufficient details, and can easily be distinguished from real images.

We believe that the in-depth connection between individual words and image sub-regions is not yet fully utilized in the current network design and the model performance could be improved upon. In fact, current frameworks build the connection on individual words and equal-sized regular-grid regions. This does not work well when multiple objects are present and multiple words are used to describe each object. For example, consider the sentence: *A man swinging a baseball bat.* We would expect the two phrases: *a man* and *a baseball bat*, each define an identifiable object in the generated image.

Therefore, in this paper, a few novel strategies have been proposed to improve the current attention based mechanism, in particular, we uniquely incorporate object-grid image features into the learning of text phrase embedding in the Encoder. This embedding is then used to compute a new set of attentions with the image in the Generator.

These strategies have delivered three unique outcomes: (1) when generating pixels inside a object-grid region, the attention is paid to the phrases rather than individual words, which makes sense from a natural language point of view. (2) pixels within each region describes the same phrase by sharing the same attention score, so that objects are more easily recognizable, (3) Moreover, the spatial information of the generated objects are more likely to be correctly reflected. The rest of this paper is organized as follows.

In section 2, we review the GAN network and several other literature that we applied as the basis and inspiration of our work. In section 3, we introduce assumptions and the architecture of our model. The performances are compared and discussed in section 4.

2. Background and Related Work

Text-to-image generation comes across multiple disciplines, and in this section we review previous methods that inspire our work.

2.1. Sentence Embedding

Generating images from text requires each sentence to be encoded into a fixed length vector. This vector is used as the conditioning factor for the image generation. Previous works such as StackGAN (Zhang et al. (2016)) and StackGAN ++ (Zhang et al. (2017)) used sentence embeddings generated by a pre-trained Convolutional Recurrent Neural Network (Reed et al. (2016b)). Recent works such as AttnGAN (Xu et al. (2017)) used a bi-directional Long Short-Term Memory (LSTM) that was trained from scratch. However, both methods were only capable of extracting word and sentence representations, which overlook the importance of phrases for the image generation.

2.2. Text-to-Image with GAN

Great progress has been achieved in text to image generation with the recent emergence of the Generative Adversarial Network (GAN) network. The GAN network was originally proposed in 2014 (Goodfellow et al. (2014)). It involves a 2-player non-cooperative game by generator and discriminator. The generator produces samples from the random noise

vector z, and the discriminator differentiates between true samples and fake samples. The value function of the game is as follows:

$$\min_G \max_D V(D,G) = \mathbb{E}_{x\sim p_{data}(x)}[logD(x|e)] + \mathbb{E}_{z\sim p_z(z)}[log(1-D(G(z)))] \quad (1)$$

Deep Convolutional GAN (DCGAN) (Radford et al. (2015)) utilized several layers of convolutional neural networks to encode and decode images, in addition to Batch Normalization (Ioffe and Szegedy (2015)) to stabilize the GAN training.

Conditional GAN (Mirza and Osindero (2014)) further allowed samples to be generated from conditioning factors. In the case of sentence based generation, the conditioning factor is a fixed length sentence embedding $\boldsymbol{e}$.

GAN-CLS (Reed et al. (2016c)) proposed one of the first works that applies conditional GAN (Mirza and Osindero (2014)) to generate plausible images. The generation is based on the sentence embedding e and a random noise vector z which is sampled from a Gaussian distribution.

Later another model GAWWN by Reed et al. (2016a) supplies additional information such as bounding boxes or part locations of the main object. Such a framework allows controllable image generation.

As previous works failed to generate images with higher-resolution than 128×128, researchers later utilized multi-stage generation process (Zhang et al. (2016)). The first stage generates a low-resolution image from the sentence embedding, and the later stages generate higher-resolution images.

AttnGAN (Xu et al. (2017)) was the first work that employs an attention mechanism between words and regular-grid regions. It was able to generate images of better quality and achieved so far the highest inception score.

Another work by Li et al. (2019) proposed object-driven image generation. This framework was a three-tier generation process. The first generated bounding boxes and corresponding labels through an attention seq2seq model. The second predicted the shape of each object through a GAN network given the bounding boxes and categories. The third was the 2-stage image generator based on the predicted shapes and labels.

2.3. RoI Pooling

The Region of Interest (RoI) pooling layer was first introduced in Girshick (2015). For an image region with spatial size $h \times w$, it is first divided into $H \times W$ grids of sub-windows. Each sub-window is then fed through a max-pooling layer, which derives a final pooling result with spatial size $H \times W$. The RoI pooling allows each image region to be embedded into a fixed-length vector with no additional parameters and training involved. In our work we use RoI pooling to extract features from the object-grid regions.

3. Architecture

Our network is inspired by several recent works including the architecture of AttnGAN as well as the visual-semantic alignment (Karpathy and Fei-Fei (2015)). Our work differs significantly, as we introduced the attentions between phrase and object-grid features into

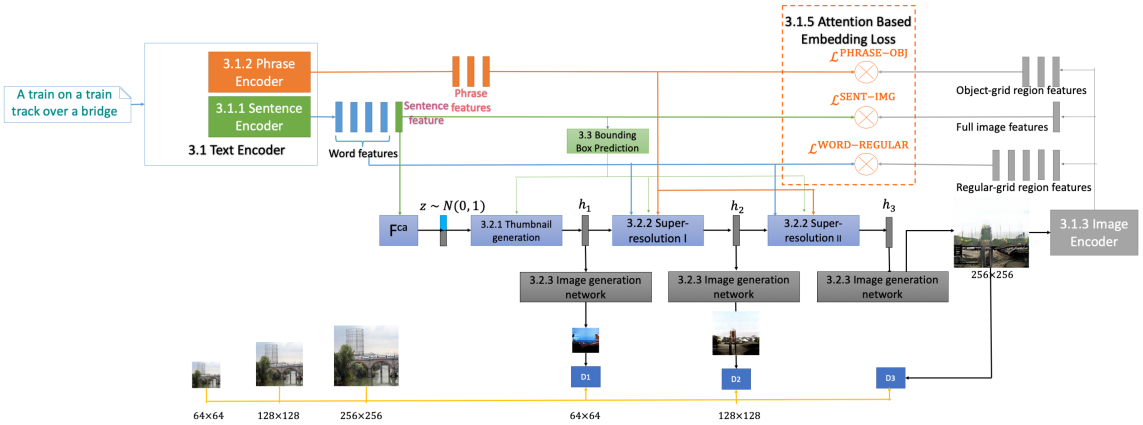


Figure 1: Network structure for text embedding and GAN network, section IDs have been added for easier reference

the network. Compared to Obj-GAN, our framework introduces the phrases and the overall design is much simpler.

To this end, we show our overall model in Figure 1 which consists of an end-to-end text encoder network, a GAN framework and a bounding boxes prediction framework. The details for each module are explained in the following sections.

3.1. Text Encoder

The text encoder of current text conditioned GAN network typically extracts a whole sentence representation and word representations using a bi-directional LSTM. In addition to those features, our proposed work also extracts the phrase features to be fed into our algorithm.

We define a phrase as a combination of the closest article (digit), adjective and noun. Such information can be extracted from applying part-of-speech tagging (POS-tagging) to raw sentences. For example, a sentence *"Two black horses standing with a cart attached to them."* is tagged as [("Two", digit), ("black", adjective), ("horses", noun), ("standing", verb), ("with", preposition), ("a", article), ("cart", noun), ("attached", verb), ("to", preposition), ("them", pronoun)]. We then group the nearest article(digit)-adjective-noun words as a phrase, which yields *"two black horses"* and *"a cart'*.

The full text encoder framework is shown in figure 2, which in fact can be considered as a phrase encoder built on top of a sentence encoder. In addition, our design incorporates object-grid image features to assist the learning. Such image features are extracted from an image encoder. The details are explained in the following sections.

3.1.1. Sentence Encoder

Firstly, a bi-directional LSTM is applied to each sentence to extract word and sentence representations. Given a sentence $\{w_1, \dots w_T\}$, the t^{th} word representation e_t is a concatenation of a forward e_t^f and a backward hidden state e_t^b, i.e., $e_t \equiv [e_t^f \; e_t^b]$. The full sentence embedding $\bar{e}$ is defined using the last hidden states, i.e., $\bar{e} \equiv [e_T^f \; e_T^b]$.

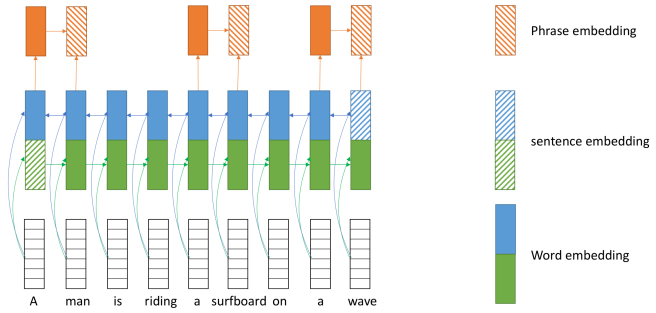


Figure 2: Text Embedding with a 2-layer LSTM Networks

3.1.2. Phrase Encoder

On top of the extracted word representations $\boldsymbol{e}$ where $\boldsymbol{e} \equiv \{e_1, \ldots e_T\}$, phrase representations are extracted by applying a second LSTM in the following way. Given the t'^{th} phrase, a LSTM is applied over the sequence of words in the phrase. The last hidden state is used as its feature representation which we refer to as $p_{t'}$. An alternative way to extract phrase embeddings is by taking the average of word embeddings in each phrase. We the performance for both methods in Section 4.2.

Our phrase-based embedding clearly has an advantage over the traditional word-based mechanism where each word has one representation. For example, none of the *individual* words in the phrase *"a green apple"* portrays an overall picture of the object; all three words work together to capture its visual meaning.

3.1.3. Image Encoder

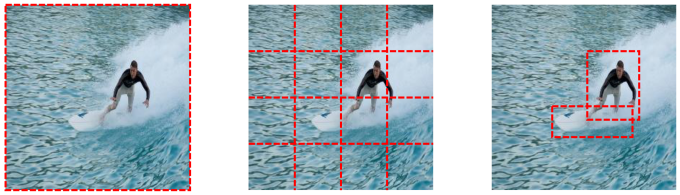


Figure 3: Examples of full image region, regular-grid region and object-grid region

The image encoder itself comes from the pre-trained Inception-v3 network (Szegedy et al. (2015)) and is not further fine-tuned in our framework. We apply the image encoder to extract three types of image features from a single image: a *object-grid region feature*, a *regular-grid region feature* and a *full image feature*. As shown in figure 3, a object-grid region is defined over a single object and thus the regions differ in sizes, and regular-grid regions have equal sizes and each of them can contain half an object or multiple objects.

Common to all features, each image first undergoes the Inception-v3 model. We use the last feature layer, i.e, "*mixed_6e*" layer as the designated layer for the regular-grid region. The full image feature is obtained from the last average pooling layer. In addition, both *regular-grid region feature* and *full image feature* are converted into vectors in the same semantic space using a trainable Fully Connected (FC) layer. Therefore, the resulting features have the following dimensions: a regular-grid region feature $v \in R^{289\times D}$ where $289 = 17 \times 17$ is the dimension for "*mixed_6e*" layer feature map. The image feature is denoted as $\bar{v} \in R^D$.

To obtain a *object-grid region feature*, the location and size of each region must first be identified. In several open datasets, such as the Microsoft COCO (MSCOCO) dataset, the manually-labeled bounding boxes of object(s) within an image are readily available. In the case where the dataset does not provide such information, they can also be obtained from off-the-shelf image object detectors, such as R-CNN (Girshick et al. (2014)). This makes it possible to apply our algorithm to any image datasets with text annotations, including the CUB and the Oxford Flower 102 datasets.

The "*mixed_6e*" layer feature map and its bounding box information is fed through the Region of Interest (RoI) pooling to generate its object-grid region feature. These features are fed through a convolution operation with a kernel of an equivalent size, resulting in a vector in a common semantic space as text features. We denote the object-grid region feature as $b \in R^{K\times D}$ where K is the number of bounding boxes in each image.

3.1.4. Attention-Based Embedding for Text

Text embedding and the perceptron layer for image and region features are bootstrapped prior to training the GAN. The training requires an overall loss function, which is defined as Equation 2:

$$\mathcal{L}^{\text{TEXT}} = \mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{WORD-REGULAR}} + \mathcal{L}^{\text{PHRASE-OBJ}} \tag{2}$$

The above loss function is comprised of three separate losses, each of them follows Xu et al. (2017). Therefore, without loss of generality, the target of the loss function is to minimize the negative log posterior probability for the correct image-sentence pair. i.e. for a batch of image-sentence pairs $(S_i, I_i)_{i=1}^M$, for clarity, we drop the subscript for $\mathcal{L}$:

$$\mathcal{L} = -\sum_{i=1}^{M}(\log P(S_i|I_i) + P(I_i|S_i)) \tag{3}$$

where $P(S_i|I_i)$ is the conditional probability for a text data S_i to be matched with an image data I_i defined as:

$$P(S_i|I_i) = \frac{\exp(\gamma_1 \mathcal{R}(S_i, I_i))}{\sum_{q=1}^{M} \exp(\gamma_1 \mathcal{R}(S_q, I_i))} \tag{4}$$

Here $\mathcal{R}(S_i, I_i)$ gives the similarity score between the text and the image data. In here, text may refer to *sentence*, *word* or *phrase*, and the image may refer to their corresponding "entire image", "regular-grid region" and "object-grid region" respectively. γ_1 is a manually

defined smooth factor. The posterior probability $P(I_i|S_i)$ for an image being matched to a sentence is defined in a similar way.

The similarity score $\mathcal{R}(S_i, I_i)$ can be defined in multiple ways using off-the-shelf methods from the statistics community to suit each situation. In our work, we apply three $\mathcal{R}$ values: $\mathcal{R}^{\text{SENT-IMG}}$, $\mathcal{R}^{\text{WORD-REGULAR}}$ and $\mathcal{R}^{\text{PHRASE-OBJ}}$ to Equation 3 and Equation 4, which derives three corresponding loss values $\mathcal{L}^{\text{SENT-IMG}}$, $\mathcal{L}^{\text{WORD-REGULAR}}$ and $\mathcal{L}^{\text{PHRASE-OBJ}}$.

3.1.5. Choices of $\mathcal{R}$ for $\mathcal{L}^{\text{SENT-IMG}}$, $\mathcal{L}^{\text{WORD-REGULAR}}$ and $\mathcal{L}^{\text{PHRASE-OBJ}}$

$\mathcal{L}^{\text{SENT-IMG}}$ describes the similarity between text and image. We have chosen $\mathcal{R}^{\text{SENT-IMG}} = \phi(S_i, I_i)$ to be the cosine similarity between a sentence representation $\bar{e}$ and a whole image feature $\bar{v}_i$.

$\mathcal{L}^{\text{WORD-REGULAR}}$ utilizes the attention mechanism built between the regular-grid regions and the words. Its similarity score is chosen as $\mathcal{R}^{\text{WORD-REGULAR}} = \log(\sum_{t}^{T} \exp(\gamma_2 \phi(c_t, e_t)))^{\frac{1}{\gamma_2}}$. In here, γ_2 is a second smooth factor. $\phi(c_t, e_t)$ is the cosine similarity between a word embedding e_t and a regular-grid-region-context vector c_t. c_t is calculated as a weighted sum over regular-grid image features: $c_t = \sum_{j=0}^{289} \alpha_j v_j$, where α_j is the attention weight for the j^{th} regular-grid towards the t^{th} word and $\alpha_j = \frac{\gamma_3 \phi(c_t, e_t)}{\sum_k^{289} exp(\gamma_3 \phi(c_t, e_k))}$.

$\mathcal{L}^{\text{PHRASE-OBJ}}$ is defined using the attention mechanism between the object-grid regions and the phrases. The similarity score is $\mathcal{R}^{\text{PHRASE-OBJ}} = \log(\sum_{t'}^{K} \exp(\gamma_2 \phi(c_{t'}, p_{t'})))^{\frac{1}{\gamma_2}}$ where $c_{t'}$ is the object-grid-region context vector. $c_{t'}$ is calculated in a similar way as c_t except that it is a weighted sum over the object-grid region features instead of regular-grid features.

Another alternative to define $\mathcal{L}$ is using the attention connection between the object-grid regions and words instead of phrases. We denote such a loss value $\mathcal{L}^{\text{WORD-OBJ}}$.

3.1.6. Overall Text Embedding Loss

Having defined the four loss values above, we applied them to our overall network architecture as shown in Graph 1. In section 4.2, we illustrated the performance of different combinations of $\mathcal{L}$. This further demonstrates that object-grid region and phrases are important conditional information in image generation.

3.2. Attentional Text to Image Generation

Inspired by previous network designs, our work constructs text to image generation as a multi-stage process. At each generation stage, images from small to large scales are generated from corresponding hidden representations. We name the first stage as "thumbnail generation" which takes sentence embedding $\bar{e}$ as the input and generates images with the lowest resolution. At the following stages, images with higher resolution are generated from the hidden state of the last stage with an attention-based structure.

3.2.1. Thumbnail Generation

The thumbnail generation is inspired by the bounding box conditioned sentence to image design by GAWWN and has been modified to suit our network design. The Generator structure is shown in figure 4.

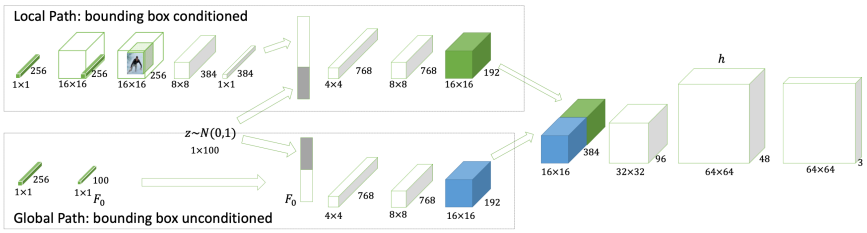


Figure 4: Thumbnail Generator

The generation process branches into two paths. The global path, which is not bounding box conditioned, takes the conditioning factor F_0 and the noise vector to produce a global feature tensor. F_0 itself is a Gaussian latent variable whose mean and diagonal covariance matrix come from F^{ca} which is a function of the sentence embedding. The local path instead uses the sentence embedding directly. It first combines the sentence embedding and the bounding box through spatially replicating e and zero out the region out of the bounding box. In the case where multiple bounding boxes coexist, the resulting tensor is averaged. The local path then takes the combined tensor through another several layers to generate a local feature tensor. Tensors from both paths are concatenated depth-wise to derive the first hidden representation h_1.

In terms of the discriminator, one naive approach to incorporate the bounding box information is to follow GAWWN, where features extracted from an image is to concatenate with the features extracted from the image-bounding boxes pair. The Discriminator then evaluates this concatenated vector. However, the experiment results shows unfavourable outcome using this approach. In our work, we introduce a three Discriminators approach. The details are discussed in section 3.2.4.

3.2.2. Super-resolution I & II

Super-resolution enlarges the previously generated thumbnails through constructing the attention mechanism between the last hidden state and text features. At stage n, a hidden representation h_n is constructed from the last hidden state h_{n-1}. h_n is later translated to an image with the *image generation network* in section 3.2.3.

We incorporate two sets of attentions in our framework. The first is between individual words and regular-grid regions, the second is between phrases and object-grid regions.

Given the word embeddings $\boldsymbol{e}$ where $\boldsymbol{e} \equiv e_1, \dots e_T$ for T words in a sentence and phrase embeddings $\boldsymbol{p}$ where $\boldsymbol{p} \equiv p_1, \dots p_{T'}$ for T' phrases in a sentence, h_n is calculated as:

$$h_n = F_n(h_{n-1}, F_n^{attn1}(\boldsymbol{e}, h_{n-1}), F_n^{attn2}(\boldsymbol{p}, h_{n-1})) \tag{5}$$

Here, F_n is a deep neural network that constructs the hidden representation h_n from given inputs. F^{attn1} and F^{attn2} are the deep neural networks that construct the word-context matrix and phrase-context matrix respectively.

The word-context matrix is constructed from word embeddings $\boldsymbol{e}$ and regular-grid image region features from h_{n-1}. $\boldsymbol{e}$ are first fed through a perceptron layer to be converted into the common semantic space as image features. The regular-grid region is defined here in a similar way to section 3.1.3, except that the input feature map is not from the pre-trained Inception-v3.

Given j^{th} regular-grid region feature h^j_{n-1}, a word-context vector c_j is defined as the weighted sum over word embeddings:

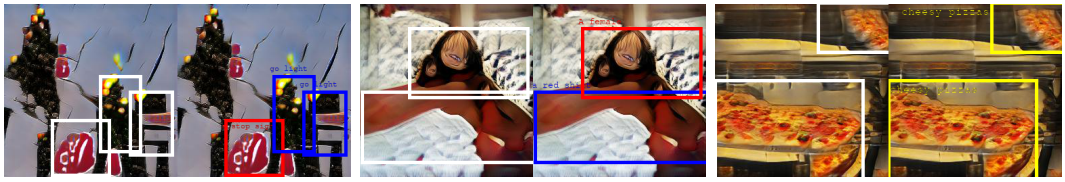
$$c_j = \sum_t^T \varphi_{j,t} e_t \tag{6}$$

Here $\varphi_{j,t}$ is the attention weight between the t^{th} word and the j^{th} regular-grid region and $\varphi_{j,t} = \frac{\exp(h^j_{n-1}{}^\top e_t)}{\sum_\tau^T \exp(h^j_{n-1}{}^\top e_\tau)}$. Suppose there are J regular-grids, the final word-context matrix is then defined as the union of the c_j value for each regular-grid region, i.e., $F^{attn1}_n(\boldsymbol{e}, h_{n-1}) = (c_1, \cdots, c_J)$.

This phrase-context matrix is calculated in a similar way, except that word embeddings are replaced with phrase features, and regular-grids are replaced with object-grid features. Here object-grid features are derived from h_{n-1} by feeding it through the RoI pooling.

The resulting phrase-context matrix is of length K where K is the number of object-grid regions defined in the image. In order to apply such a matrix to the network, we let each pixel inside the bounding box carry the same phrase context vector while pixels outside of bounding box carry zeros. As for regions where multiple bounding boxes overlap, the phrase context vectors are averaged. Thus, the resulting phrase-context matrix is of the same shape as the previously defined word-context matrix.

In Figure 5 we show examples of attention-weights mapping, from object-grid image regions to phrases. It is clear to see that our framework encourages the correct text information to be focused in generating each key objects.



(a) A picture of a stop and go light with a stop sign next to it.

(b) A female wearing a red shirt lies on a bed, resting.

(c) A metal counter topped with lots of cheesy pizzas.

Figure 5: Example of attention being paid to a phrase when generating each object-grid region. White rectangles on the left figure highlight the object-grid regions in the image. The matched pair of phrase and object-grid image region is highlighted in the right figure.

3.2.3. Image Generation Network: Hidden Representation to Images

As shown in Figure 1, the previous thumbnail generation and super-resolution stages do not produce images directly. They instead produce hidden representations that are fed through an additional convolution layer using kernel size and a depth dimension 3 to generate images.

3.2.4. Discriminators

In general, we use three types of Discriminators. The first evaluates an entire image as being real or fake, which we named it D^{im}. The second evaluates a pair of image and sentence, we named it $D^{\text{im-txt}}$. The third evaluates a group of image, sentence and bounding boxes, we named it $D^{\text{im-txt-bnd}}$. Collectively, our Discriminator set is: $\mathcal{D} \equiv \{D^{\text{im}}, D^{\text{im-txt}}, D^{\text{im-txt-bnd}}\}$

In addition, we incorporate the logic of matching-aware Discriminator from Reed et al. (2016c), where the latter two Discriminators are fed through real, fake and unmatched samples. The value function for the Generator and the Discriminator at each stage is given below where we denote the bounding box information as b:

$$\begin{aligned}\min_{G}\max_{\mathbf{D}} V(\mathcal{D}, G) = &\mathbb{E}_{x_i \sim p_{data}(x_i)}[\log \mathcal{D}(x_i)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(G(z|\bar{e})))] \\ &+ \mathbb{E}_{x_i \sim p_{data}(x_i)}[\log \mathcal{D}(x_i, \bar{e})] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(G(z|\bar{e}), \bar{e}))] \\ &+ \mathbb{E}_{x_i \sim p_{data}(x_i)}[\log \mathcal{D}(x_i, \bar{e}, b)] + \mathbb{E}_{z \sim p_z(z)}[\log(1 - \mathcal{D}(G(z|\bar{e}), \bar{e}, b))]\end{aligned} \tag{7}$$

In Table 1 we report the detailed network architecture for the discriminator performed on the smallest 64×64 images. $D_d = 96$, $D_e = 256$ are the chosen hyper-parameters. f_D refers to features produced in the network. Up-sampling consists of a nearest neighbour image resize, a convolution, a batch normalization and a GLU layer. Down-sampling consists of a convolution, a batch normalization and a leaky ReLU layer. The kernel size and the stride value used in both operations are shown in the bracket. Note that identical function/framework applies to all stages with deeper network designs on larger images. The full network architecture can be found in the Appendix.

Stage	Sub-stage	Name	Input Tensors	Output Tensors
Image + Sentence Discriminator		Convolution + leaky ReLU	$64 \times 64 \times 3$	$32 \times 32 \times D_d$
		Down-sampling (kernel=4, stride=2) $\times 3$	$32 \times 32 \times 96$	$f_D^{IMG}(4 \times 4 \times (D_d \times 8))$
	$\mathcal{D}(x)$	Convolution (Image only logits)	$4 \times 4 \times (D_d \times 8)$	1
	$\mathcal{D}(x, \bar{e})$	\textbf{Sentence conditioned logits}	f_D^{IMG}, $\bar{e}$	1
Image + Sentence + Bounding Box Discriminator		Convolution	$64 \times 64 \times 3$	$32 \times 32 \times D_d$
		Down-sampling (kernel=4, stride=2)	$32 \times 32 \times D_d$	$f_D^{IMG^2}(16 \times 16 \times (D_d \times 2))$
		Spatial Replicate	$\bar{e}$	$16 \times 16 \times D_e$
		Concatenation	$16 \times 16 \times D_e$, $f_D^{IMG^2}$	$16 \times 16 \times (D_e + D_d \times 2)$
		Apply bounding box mask	$16 \times 16 \times (D_e + D_d \times 2)$	$16 \times 16 \times (D_e + D_d \times 2)$
		Down-sampling (kernel=4, stride=2) $\times 2$	$16 \times 16 \times (D_e + D_d \times 2)$	$f_D^{IMG-BBOX}(4 \times 4 \times (D_d \times 8))$
	$\mathcal{D}(x, \bar{e}, b)$	\textbf{Sentence conditioned logits}	$f_D^{IMG-BBOX}$, $\bar{e}$	1
Sentence conditioned logits		Spatial replicate	$\bar{e}$	$4 \times 4 \times D_e$
		Concatenation	f_D^{IMG} or $f_D^{IMG-BBOX}$, $\bar{e}$	$4 \times 4 \times (D_e + D_d \times 8)$
		Down-sampling (kernel=3, stride=1)	$4 \times 4 \times (D_e + D_d \times 8)$	$4 \times 4 \times (D_d \times 8)$
		Convolution	$4 \times 4 \times (D_d \times 8)$	1

Table 1: Network Architecture for the Discriminators on the 64×64 images.

3.3. Bounding Box Prediction

As the image generation relies on bounding box information which is not available in the testing phase, a separate bounding box prediction network is trained based on the sentence

embedding. We define two prediction tasks in the network. The first is to predict the coordinates for bounding boxes. The second is to predict the total number of bounding boxes described in the sentence. We structure both prediction as a regression problem. Therefore, given a sentence embedding, it is first fed through 2 multi-layer neural networks, in which is the final layer of both networks is a mean squared error of the predicted value and the real value.

We adopt several processing steps on the data in the following manner. First, the coordinates of bounding boxes is normalized to the proportion of the full length, so that the maximum value is 1 regardless of the size of the bounding box or the image. Second, given a predicted number of bounding boxes, the coordinates for the bounding boxes that outnumber the predicted value are considered "invalid", and are thus excluded in computing the loss. In addition, we define words such as "left", "right" as position related words. We later compare the performance between using all sentences for the training and using only sentences that contain position related words.

4. Experiments

Below we demonstrate the superior performance of our work, as well as the performance of each of the proposed components, i.e., the text encoder, the GAN network and the bounding box predictor from sections 3.1, 3.2 and 3.3 respectively.

The dataset we used is the MSCOCO dataset, which includes various images that involve natural scenes and complex object interactions. It contains 82,783 images for training and 40,504 for validation. Each image has 5 corresponding captions. Bounding boxes are provided for objects in 80 categories.

The text encoder is trained over 150 iterations and the learning rate is set as 0.0002. Then the fine-tuned text encoder is used to train the GAN network over 120 iterations and the learning rate for both the generator and the discriminator is set as 0.0002.

Three metrics, Inception scores, Frechet Inception Distance (FID) score and R-precision were utilized to perform the evaluation.

4.1. Metrics: Inception Score, FID Score and R-precision

As it is difficult to measure the performance of image generation in a quantitative way, the *inception score* (Salimans et al. (2016)) and the *Fréchet Inception Distance* (FID) score (Heusel et al. (2017)) were two popular metrics for automatic image quality evaluation.

Both scores measure only the quality and diversity of images generated, but not how accurate an image can reflect the description of a sentence. Therefore, a third metric called R-precision is used in the previous work (Xu et al. (2017)).

R-precision is defined as the top r relevant text descriptions out of R retrieved texts for an image, and the candidate sentences are one relevant and 99 randomly selected sentences. We observed through experiments, that when the sample size (i.e., number of candidate sentence) is small, the R value has very high variance. Therefore, we used two sample sizes at 100 and $30,000$, which we named R-precision(100) and R-precision(30K) respectively.

Limitations of both Inception and FID score were pointed out in several previous literature (Barratt and Sharma (2018), Lucic et al. (2018)). In terms of R-precision, it requires

a pre-trained image-to-text retrieval model that is not available in some previous methods. Therefore, we also show examples of our generation results versus previous literature.

The experiment results demonstrated below were performed on 30,000 random samples from the validation set for the IS score and the R-precision values. The FID score is reported over the full validation set.

4.2. The Text Encoder

Below in Table 2, we demonstrate the performance of multiple text embedding losses introduced in Section 3.1.4 and Section 3.1.5. The comparison is made in terms of the final R-precision scores on the testing set. When calculating the R-precisions, the relevance between a pair of an image and a sentence is calculated using the cosine similarity between the full feature of both.

Table 2 shows that applying $\mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{PHRASE-OBJ}}$ achieves the highest R-precision scores, which are 0.07% higher testing R-precision(100) and 0.12% higher R-precision(30K) than the baseline model. The score is also higher than the two experiments that applying word-regular-grid attention (i.e. using $\mathcal{L}^{\text{WORD-OBJ}}$) to the learning. This shows that firstly, using solely phrase and object-grid regions to construct the encoder loss is better than applying $\mathcal{L}^{\text{WORD-REGULAR}}$ and $\mathcal{L}^{\text{PHRASE-OBJ}}$ together; secondly, introducing phrases is important in learning the text representation.

Experiment	R-precision(100)(%)	R-precision(30K)(%)
$\mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{WORD-REGULAR}}$ (baseline)	72.99 ± 4.50	4.732 ± 0.018
$\mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{WORD-REGULAR}} + \mathcal{L}^{\text{PHRASE-OBJ}}$	72.39 ± 4.26	4.481 ± 0.005
$\mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{WORD-OBJ}}$	71.91 ± 1.83	4.473 ± 0.017
$\mathcal{L}^{\text{SENT-IMG}} + \mathcal{L}^{\text{WORD-REGULAR}} + \mathcal{L}^{\text{WORD-OBJ}}$	70.69 ± 2.99	4.030 ± 0.013
$\mathcal{L}^{\textbf{SENT-IMG}} + \mathcal{L}^{\textbf{PHRASE-OBJ}}$	$\mathbf{73.06 \pm 4.05}$	$\mathbf{4.851 \pm 0.015}$

Table 2: The R-precision(100) and R-precision(30K) on the testing set for the text encoders. LSTM-BASIC is the basic bi-LSTM used in AttnGAN which is our baseline model. LSTM and LSTM-PHRASE comes from the proposed method.

4.3. The GAN Network

Table 3 reports the R-precision, Inception score (IS) and FID score achieved by the previous algorithms and the proposed methods. Apart from the metrics for the proposed method, the R-precision(30K) score and the FID score for AttnGAN, other scores came from previous literature (Zhang et al. (2017)). As StackGAN did not provide a way to extract image features, the R-precision values were not reported.

We denote the proposed method which embeds the text information with LSTM-PHRASE in addition to utilizing the object-grid regions and phrases in constructing the attention mechanism as the method Proposed. In order to demonstrate the importance of introducing phrase and object-grid attention, Table 3 also reports the result for first, using only word-regular-grid attention and second, using word-regular-grid and word-object-grid attention. We denote these two as WORD-REGULAR and WORD-REGULAR+WORD-OBJ respectively. Both of them use the same text embedding as the *Proposed* method.

The proposed method achieves 4.2% higher R-precision (100) and 4.1% higher R-precision (30K) than the baseline, which shows that the proposed method is able to generate im-

ages that match more closely with the content described in the sentence. In addition, the proposed method also achieves better performance than WORD-REGULAR and WORD-REGULAR+WORD-OBJ. This shows that it is important to introduce phrases to the learning.

Method	R-precision(100)(%)	R-precision (30K)(%)	IS(30K)	FID
StackGAN-v1			8.45 ± 0.03	74.05
StackGAN-v2			8.30 ± 0.10	81.59
AttnGAN	85.47 ± 3.69	6.72 ± 0.15	25.89 ± 0.47	32.12
WORD-REGULAR	85.97 ± 3.01	8.20 ± 0.16	26.18 ± 0.30	40.54
WORD-REGULAR+WORD-OBJ	88.25 ± 3.01	10.37 ± 0.13	24.81 ± 0.21	36.51
Proposed	89.69 ± 4.34	10.80 ± 1.96	26.92 ± 0.52	34.52

Table 3: R-precision, Inception and FID score score between AttnGAN and the proposed method.

Below we show two aspects with real samples that the generation result surpasses previous methods.

Firstly, as shown in Figure 4, it is able to generate images that match closer with a given sentence, which is proven by the higher R-precision rate. This means that the proposed method is less likely to "miss" objects. For example. In the case where "stop sign" and "go light" are both mentioned, the proposed method is able to generate both objects instead of only focusing on the "stop sign". In addition, when multiple objects / object-grid regions of the same type co-exist in the image, the proposed method is able to generate the correct number of objects.

Secondly, the proposed method performs specifically well in displaying identifiable main object, such as "A female" or "Large brown cow". Through feeding the true-gird region information, the proposed method is able to focus the attention on a more precise image region instead of the entire image.

4.4. Bounding Box Prediction

In the phase of validation and training, the coordinates of each bounding box and number of bounding boxes are predicted by separate networks, as explained in section 3.3. In this section, we compare the performances of multiple alternatives of both predictions. Below we denote the two prediction tasks as "Coordinates Prediction" and "Number Prediction" respectively. Comparisons were made in terms of the validation losses over the training iterations.

The first comparison is whether to use all sentences in the prediction tasks or to only use those sentences with position related words. The second is in terms of the number of layers used for the prediction

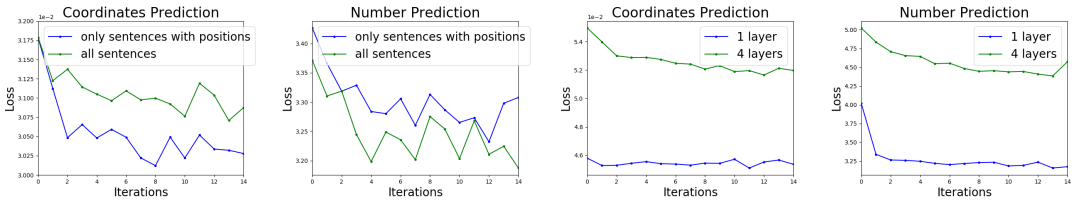


Figure 6: Validation losses of coordinates prediction and number prediction in terms of whether to use the sentences that include position related words for the training.

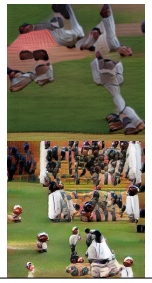
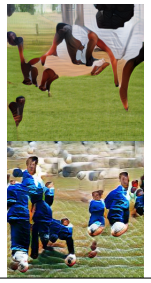
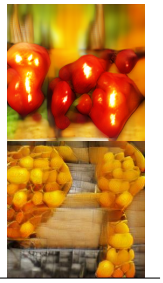
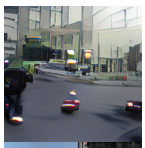
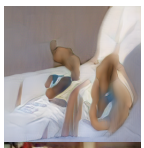
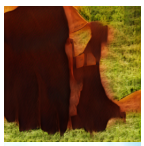
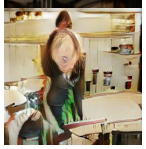
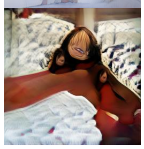
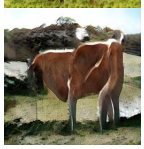
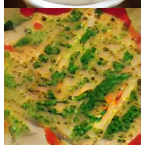
Caption	Baseball players run after a ball during a game	A picture of a stop and go light with a stop sign next to it	A kitchen filled with wooden cabinets and a microwave oven.	A metal counter topped with lots of cheesy pizzas.	A group of young men standing on top of a soccer field.	A pile of oranges sitting inside of a basket.
AttnGAN						
Proposed						
Caption	A woman in white shirt standing in kitchen area.	A busy traffic area on a street during the day	A female wearing a red shirt lies on a bed, resting	Large brown cow standing in field with small cow	A pizza with purple cabbage topping on a table next to white bowl	Black and white photo of a pedestrian at a suburban crosswalk
AttnGAN						
Proposed						

Table 4: Examples of images generated by the proposed method and AttnGAN.

From Figure 6 , it is clear to see that the best practice is applying 1 layer neural network to both predictions, and that the out-numbered bounding boxes should be excluded when calculating the losses.

5. Summary

Our work provides improvements over the state-of-the-art attention based GAN network for text to image generation. Using *phrase* as an additional important encoding unit into image generation, our proposed work has incorporated the following two innovations: Firstly, we proposed a new design of text embedding which extracts additional phrase embedding. Secondly, we incorporate a new set of attentions computed between object-grid regions and phrases and bring them into our GAN network design. Through the experimentation on

the MSCOCO dataset, our approach is capable of generating more realistic and accurate images.

References

Shane Barratt and Rishi Sharma. A note on the inception score. *arXiv preprint arXiv:1801.01973*, 2018.

Ross Girshick. Fast r-cnn. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2672–2680. Curran Associates, Inc., 2014. URL http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500, 2017. URL http://arxiv.org/abs/1706.08500.

Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015. URL http://arxiv.org/abs/1502.03167.

Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

Wenbo Li, Pengchuan Zhang, Lei Zhang, Qiuyuan Huang, Xiaodong He, Siwei Lyu, and Jianfeng Gao. Object-driven text-to-image synthesis via adversarial training. *CoRR*, abs/1902.10740, 2019. URL http://arxiv.org/abs/1902.10740.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 700–709. Curran Associates, Inc., 2018. URL http://papers.nips.cc/paper/7350-are-gans-created-equal-a-large-scale-study.pdf.

Mehdi Mirza and Simon Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014. URL http://arxiv.org/abs/1411.1784.

M-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. *CoRR*, abs/1511.06434, 2015. URL http://arxiv.org/abs/1511.06434.

Scott E Reed, Zeynep Akata, Santosh Mohan, Samuel Tenka, Bernt Schiele, and Honglak Lee. Learning what and where to draw. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 217–225. Curran Associates, Inc., 2016a. URL http://papers.nips.cc/paper/6111-learning-what-and-where-to-draw.pdf.

Scott E. Reed, Zeynep Akata, Bernt Schiele, and Honglak Lee. Learning deep representations of fine-grained visual descriptions. *CoRR*, abs/1605.05395, 2016b. URL http://arxiv.org/abs/1605.05395.

Scott E. Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *CoRR*, abs/1605.05396, 2016c. URL http://arxiv.org/abs/1605.05396.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, Xi Chen, and Xi Chen. Improved techniques for training gans. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems 29*, pages 2234–2242. Curran Associates, Inc., 2016. URL http://papers.nips.cc/paper/6125-improved-techniques-for-training-gans.pdf.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *CoRR*, abs/1512.00567, 2015. URL http://arxiv.org/abs/1512.00567.

P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, California Institute of Technology, 2010.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *CoRR*, abs/1711.10485, 2017. URL http://arxiv.org/abs/1711.10485.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. *arXiv preprint arXiv:1612.03242*, 2016.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *CoRR*, abs/1710.10916, 2017. URL http://arxiv.org/abs/1710.10916.