

# Text to Image Synthesis with Mutual Information Optimization

**Lihang Liu**  
UT Austin

lihangliu@utexas.edu

**Mei Wang**  
UT Austin

meiwang@utexas.edu

## Abstract

Text to image synthesis is an interesting and useful task to explore recently. The goal is to generate images from captions, i.e. taking unstructured textual descriptions as input and using them to generate relevant images. This task combines two challenging components of language modeling and image generation and can be considered to be more difficult than caption generation. In this paper, we proposed a method that combines both the pipelines of text to image synthesis and image captioning to help maximize the information flow through these pipelines and improve the quality of generated images. In our experiments on CUB and Oxford-102 datasets, we demonstrate the effectiveness of our method on generating images of specific categories from unstructured text descriptions.

## 1 Introduction

Recently with the help of deep neural networks, impressive progress has been made on text to photo-realistic image synthesis (Kingma and Welling, 2013; Goodfellow et al., 2014; Gregor et al., 2015; Denton et al., 2015). Among all the state-of-art networks for text-to-image synthesis, Generative Adversarial Networks (GAN) shows its advantages in generating sharper images. Plenty of techniques have been proposed to further improve GAN for more stable training process and higher resolution images (Salimans et al., 2016; Reed et al., 2016b; Yang et al., 2017; Zhang et al., 2017a,b; Chen et al., 2018).

However, their main focus is to generate images with higher resolution and photo-realistic, and none of them ensures that the information in the input text is expressed in the generated image in a complete manner, which means it may be hard to infer the original text from the generated image. Instead of mainly focusing on improving the resolution, the completeness of info from the input text



Figure 1: Comparison of Stack GAN with and without our proposed information flow on  $64 \times 64$  images. The top two images are generated from Stack GAN and the bottom two images are generated from our proposed method. Our method better conserves the information from the input sentence.

should be a more fundamental task that needs to be guaranteed first. Let's describe this problem in information theory. Given an input sentence with words describing some images, we use a generator to obtain a fake image, which is the essential component of the text to image synthesis pipeline. To encourage the fake image to express the information of the input sentence as much as possible, we want to maximize the mutual information of the input sentence and the fake image. However, directly computing the mutual information is intractable, so we need to propose another new neural network to approximate this metric and optimize with it so as to produce high-quality images with more complete info from the text.

In this paper, we combine the pipelines of text to image synthesis and image captioning to maximize the information flow based on GAN network,

such that more complete info is encouraged in the training pipelines. Compared to traditional text to image synthesis, after adding the image captioning pipeline, we can generate a fake sentence from the fake image and feed it back to the generator. More concretely, we design three components in our network. Firstly, we use a *Generator* to generate the fake image from the input text. Secondly, we use a *Discriminator* to judge whether the generated image is real or fake. Thirdly, we use a *Captioner* component to generate the text from the generated image. The first two components are the standard GAN framework (Goodfellow et al., 2014). While the third component Captioner is used to recover the text given the generated image, which provides another signal to train the Generator. The major advantage of the third component is that we can require the generated image to contain as much information of the input text as possible and this can further improve the quality of the generated image given extra signals iteratively. That is, with this new Captioner component, we can approximately maximize the mutual information of the input sentence and the fake image, such that more complete information is encouraged in the pipelines which will produce higher quality images. For example, in Figure 1, we want to generate bird figures as the text captions described on the top. However, the top two figures generated from StackGAN do not show some primitive properties like gray nape, darker wings, etc. While after adding our mutual info optimization design, the bottom two figures give a better illustration of the texts.

The contribution of this paper is threefold. First, it’s novel to propose the idea of maximizing information flow in the task of text to image synthesis, and we use an approximation method to solve this problem. Second, we design a new network component in GAN architecture to complete the information flow and implement four variants of objective functions Third, qualitative analysis and user study of our experiments prove the effectiveness and usefulness of our design.

The rest of this paper is arranged as follows. In Sec.2, we describe related works on text to image synthesis. In Sec.3, we introduce our methodology, which includes the background of GAN and our model architecture. In Sec.4, experimental analysis and user study demonstrate the effectiveness of our method. Finally, we conclude our work in Sec.5.

## 2 Related Works

Tons of recent success of deep neural network has been achieved in various tasks, typically in NLP and Computer Vision areas. Prior works on text to image synthesis are mostly based on recent popular algorithms on language modeling and deep generative models.

With the emergence of deep learning, Variational Autoencoders (VAE) (Kingma and Welling, 2013) first formulated this problem with probabilistic graphical model and continuous latent variables whose goal was to maximize the lower bound of data likelihood. Later another approach named DRAW (Gregor et al., 2015) built a Deep Recurrent Attention Writer which further incorporated a novel differentiable attention mechanism. Text modality and text feature representation are challenging subproblems. AlignDRAW (Mansimov et al., 2015) used an attention model to iteratively generate patches on canvas. They have the advantage that focuses on different phrases each time step. However, their generated images are far from photo-realistic and have low resolution.

Recently, Generative Adversarial Networks (GAN) (Goodfellow et al., 2014) shows its advantages in generating sharper image, and many modified and extended works have been proposed for further optimization. Given the unstable training dynamics of GAN, several techniques (Salimans et al., 2016; Arjovsky and Bottou, 2017) have been proposed to stabilize the training process and generate compelling results. Conditional GANs (cGANs) use conditional information for the discriminator and generator and they have drawn promising results for image generation from text (Mirza and Osindero, 2014; Denton et al., 2015; Reed et al., 2016a). Impressively (Reed et al., 2016b) is able to generate photo-realistic images with the help of GAN. Also, they proposed several variants by using different objective functions to enforce smoothness on the language manifold and reduce overfitting. Based on that, StackGAN (Zhang et al., 2017a,b) is the first model to use conditional loss for text to image synthesis, and it is one of the latest works that can synthesize high-quality images from coarse to fine in  $256 \times 256$  from text descriptions.

Our design is based on GAN architecture and we choose StackGAN as baseline method. Instead of focusing on generating high-resolution and photo-realistic images, we add another com-

ponent to maximize the mutual information of the input text and the generated image to express as much information as possible.

### 3 Methodology

In this section, we first briefly describe some preliminaries of GAN which we built upon, and then introduce our new network architecture design.

#### 3.1 Background

Generative Adversarial Networks (GAN) is the class of unsupervised learning, which is composed of two models: the generator model  $G$  and the discriminator model  $D$ . These two models are optimized alternatively. The discriminator  $D$  is optimized to distinguish whether an image is a real image or fake image that generated from  $G$ , while the generator  $G$  is optimized to generate images that are close to the true distribution of the real images such that  $D$  can't distinguish.

More concretely, we optimize the following objective function which is similar to a two-player min-max game:

$$\min_G \max_D V(D, G) = E_{x \sim p_{data}} [\log D(x)] - E_{z \sim p_z} [\log(1 - D(G(z)))] \quad (1)$$

where  $x$  is sampled from the true data distribution  $p_{data}$  and  $z$  is a noise vector sampled from Gaussian distribution  $p_z$ . It is proved in (Goodfellow et al., 2014) that this min-max game has global optimum when  $p_g$  converges to  $p_{data}$ .

#### 3.2 Information Flow

The task of text to image is to generate the image from the text description and we want to maximize the information of the input text to be expressed in the generated image. The GAN model is able to generate photo-realistic images, however, we are not sure whether the information from the input text is expressed in the generated image as much as possible, since the information is learned in a implicit way. Also, it's well known that GAN models suffer from mode collapse, which cast further concerns about whether the information is well reserved.

To solve this limitation, we introduce the idea of information flow. That is, given the input text, we want the information from the text to be reserved and expressed as much as possible in the generated image. In other words, we want to maximize the

mutual information of the input text and the generated images. However, it's intractable to directly calculate the mutual information of the text and the images. To approximate this problem, we can make an assumption that if the generated image reserves the information from the text, then from the generated image, we should also be able to recover the original text from the generated image. This can be expressed in the chain manner:

$$text \rightarrow image \rightarrow text.$$

In addition to text to image pipeline, the generated image can be decoded back to text description or embedding vector and use this info to maximize the mutual info in objective function. In this way, we realize combining both pipelines of text to image synthesis and image captioning in the information flow.

#### 3.3 Model Architecture

The network architecture are shown in Fig. 2. There are 3 main components in our network architecture (best view in colors):

- Generator  $G$ : the generator takes the text description and a random noise  $z$  from Gaussian distribution as input and generate a fake image.
- Discriminator  $D$ : the discriminator takes a fake or real image as input along with the corresponding text embedding  $\psi$ , then judge whether the input image is fake or real.
- Captioner  $C$ : the Captioner will take the fake or real image as input and try to decode into a feature vector that matches the size of the text embedding.

The Generator and Discriminator can be summarized into a traditional text to image synthesis work (Mansimov et al., 2015; Reed et al., 2016b; Zhang et al., 2017a) using GAN techniques. In the experiment, we use the implementation of StackGAN as our baseline. While the Captioner is used to enforce the information flow introduced in Sec.3.2. The naming comes from the similarity to the traditional Image Captioning works (Karpathy and Fei-Fei, 2017; Collobert and Collobert, 2015) that map the images into text descriptions.

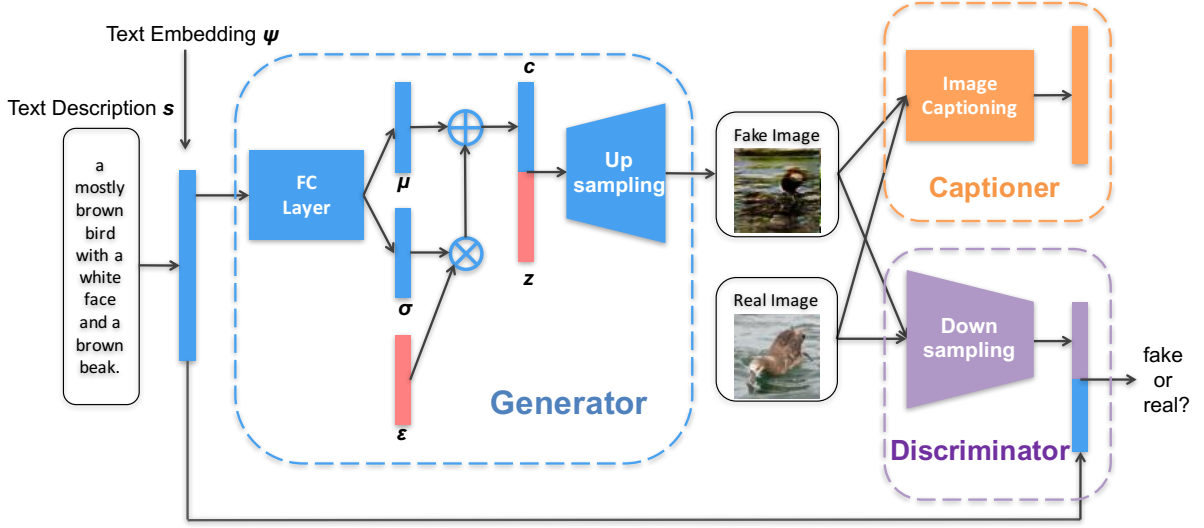


Figure 2: Network Architecture. There are 3 main components: the Generator  $G$  in blue, the Discriminator  $D$  in purple and the Captioner  $C$  in orange.  $G$  and  $D$  are the standard components of GAN, while  $C$  is our proposed components to implement the information flow, which takes the fake or real image as input and output a feature vector.

### 3.3.1 Text to Image Synthesis

In this subsection, we will introduce the network framework that we used for text to image synthesis. Basically, any framework on text to image synthesis can be extended by our algorithm. Here, we follow the ideas from VAE (Kingma and Welling, 2013) and GAN (Goodfellow et al., 2014).

More concretely, for a given text description  $s$ , we obtain its corresponding text embedding  $\psi$  using pretrained text encoding model. Then we use a fully-connected layer to generate two vectors  $\mu$  and  $\sigma$ , following the reparameterization trick introduced in Variational Auto-encoder (VAE) (Kingma and Welling, 2013). Finally we use the following element-wise formula to get the latent text vector  $c$ :

$$c = \mu + \sigma * \epsilon \quad (2)$$

where  $\epsilon$  is a noise vector sampled from the normal distribution  $N(0, 1)$ . The motivation of introducing  $\epsilon$  is to enforce smoothness in the latent text space.  $z$  is also a random noise vector sampled from the normal distribution, which is a standard setting used in GAN. By varying  $z$ , we can generate different images based on the input text. Then  $c$  is concatenated with noise vector  $z$  and generate fake images by upsampling blocks. For the discriminator  $D$ , the text embedding  $\psi$  is fed with downsampled real image vectors. Finally, the Discriminator can produce the decision, fake or real.

### 3.3.2 Captioner

In the information flow chain  $text \rightarrow image \rightarrow text$  introduced in Sec.3.2, the text to image synthesis serves as the first half  $text \rightarrow image$ , while the Captioner component serves the  $image \rightarrow text$  part. Basically, any framework of Image Captioning works can be extended with our algorithm design.

The Captioner used in our experiments shares the same architecture with the Discriminator. It takes a fake or real image as input and use a basic Convolutional Network to encode the image into a feature vector  $\psi'$ . So our target goal is to make the imaged decoded text feature vector  $\psi'$  similar to the original text embedding vector  $\psi$ . The optimal  $\psi'$  should contain the same information as the text embedding  $\psi$ . More concretely, we want to maximize the mutual information between  $\psi$  and  $\psi'$  as shown below:

$$I(\psi; C(G(\psi))) \quad (3)$$

However, we cannot optimize the mutual information directly since it is intractable in our problem. Thus in Sec.3.4, we introduce a new objective function to approximate it.

Note that, to save computation, we also share weights between the Captioner and the Discriminator. However in our theoretical analysis, the Discriminator  $D$  and Captioner  $C$  are considered independently.



### 3.4 Objective Functions

Directly maximizing the mutual information is intractable, instead we approximate this problem by minimizing the Euclidean distance between the text embedding  $\psi$  and  $\psi'$ . Other approximation methods can be further implemented in future works.

The three components in Sec.3.3 can be optimized alternatively. Specifically speaking, we denote  $(I_t, t)$  as the real image and the input sentence pair sampled from the real data distribution  $p_{data}$ , with  $\psi_t$  as the corresponding text embedding. Following the formulas of GAN, we alternatively train the Generator, the Discriminator and the Captioner by optimizing the following formula:

$$\min_{G,C} \max_D L_G + L_D + L_C \quad (4)$$

where  $L_G, L_D, L_C$  represents for the loss function of the Generator, the Discriminator and the Captioner respectively. To be exact,

$$L_G = E_{z \sim p_z, t \sim p_{data}} [\log(1 - D(G(z, c)), \phi_t)] + \lambda D_{KL}(N(\mu(\phi_t), \Sigma(\phi_t)) || N(0, I)) \quad (5)$$

$$L_D = E_{(I_t, t) \sim p_{data}} [\log D(I_t, \phi_t)] + E_{z \sim p_z, t \sim p_{data}} [\log(1 - D(G(z, c)), \phi_t)] \quad (6)$$

$$L_C = E_{z \sim p_z, t \sim p_{data}} [e(C(G(z, c)), \phi_t)] + E_{(I_t, t) \sim p_{data}} [e(C(I_t), \phi_t)] \quad (7)$$

in which the  $e$  function stands for the mean square error between  $C(G(z, c))$  and the text embedding.  $D_{KL}$  is the KL divergence regularization term used to enforce the distribution of  $c$  to be a normal distribution from VAE.

Overall,  $G$  is trained to generate fake images from the text description that are closed to the real data distribution, and it should get to the minimum of the Euclidean distance between  $\psi_t$  and  $C(G(\psi_t))$ .  $D$  is trained to distinguish the fake and real images. And  $C$  is trained to minimize the Euclidean distance between  $\psi_t$  and  $C(G(\psi_t))$  and the distance between  $\psi_t$  and  $C(I_t)$ . In this way, our new network cooperating with three components can learn to reserve and express more mutual information between text descriptions and images.

## 4 Experiments

To validate our approach, we first analyze the qualitative results with figures generated by five methods for different primitive properties, and then conduct human evaluation as well. The baseline model is generated by the first stage of StackGAN using the existing code they released online. We compare four variant models listed in Sec.4.3 with it and demonstrate that our models produce higher-quality samples than StackGAN and express more correct info in the generated figures.

### 4.1 Dataset

To evaluate our method, we conduct the experiments on two dataset: CAtech-UCSD Birds (CUB) (Wah et al., 2011) dataset and Oxford-102 Flowers (Nilsback and Zisserman, 2008) dataset. The CUB dataset contains 200 bird categories with 11,788 images. Since each image is associated with a bounding box, we preprocess the image by cropping its bounding box to ensure the object-image size ratio greater than 0.75. While the Oxford-102 dataset consists of 102 flower categories with 8,189 images.

### 4.2 Experimental Setup

Our network architecture design is based on StackGAN (Zhang et al., 2017a). In StackGAN, their Stage-I sketches the primitive properties like shape and color, and Stage-II improves its resolution and refines with details. Since in current stage, we just want to purely demonstrate the effectiveness of our new proposed third component *Captioner* in our model design based on GAN. For the sake of brevity, we only adopt the Stage-I of the Stack GAN as the baseline. In the following contents, we use StackGAN to refer to the Stage-I of StackGAN paper. Stack GAN is based on GAN techniques. In our method, we mix the StackGAN with our proposed Captioner component.

The size of text embedding  $\psi$  is 1024. And we use Adam (Kingma and Ba, 2014) optimizer throughout the experiments with batch size of 64 and the initial learning rate is 0.0002. For every 100 epochs, the learning rate is decayed by 0.5 times. During test phase, for each sentence, we sample 4 different Gaussian noise  $z$  to get 4 different fake images.

Description	Groundtruth	StackGAN	StageInfo	StageInfo_dfake	StageInfo_dfake_embed	StageInfo_dfake_dreal_embed
The bird is almost dark gray with long skinny wings and white head.						
A bird has a large hooked bill, long neck, brown and white feathers and white crown.						
A black bird with long tail feathers and large curved beak.						
The bird is completely black back which has tiny brown bill and skinny body.						

Figure 3: Example results on CUB dataset by StackGAN StageI and our improved methods, i.e. StageInfo, StageInfo\_dfake, StageInfo\_dfake\_embedded and StageInfo\_dfake\_dreal\_embed.

### 4.3 InfoStack Variants

To evaluate the effectiveness of our proposed method, we compare 4 different variants with different  $L_C$ :

- First, **StageInfo** method directly use the feature vector of  $D$  as the  $\psi'$  and compute its distance to the text embedding  $\psi$ :
- Second, **StageInfo\_dfake** method is our first try to add *Captioner* model to StackGAN with the fake image info generated by  $G$ . Basically, we change the objective function with Captioner Loss function  $L_C$  on the latent code  $c$ :

$$L_C = E_{z \sim p_z, t \sim p_{data}} [e(C(G(z, c)), c)]$$

- Third, **StageInfo\_dfake\_embed** method calculates the loss of embeddings rather than the latent code  $c$ :

$$L_C = E_{z \sim p_z, t \sim p_{data}} [e(C(G(z, c)), \phi_t)]$$

- Finally, the **StageInfo\_dfake\_dreal\_embed** method further add the real image info based on StageInfo\_dfake\_embed method:

$$L_C = E_{z \sim p_z, t \sim p_{data}} [e(C(G(z, c)), \phi_t)] + E_{(I_t, t) \sim p_{data}} [e(C(I_t), \phi_t)]$$

### 4.4 Evaluation Metrics

It's hard to compare the quality of generated images by some handcrafted rules. To analyze the results quantitatively, we ask people outside the project to visually evaluate images generated from the baseline and from our proposed method.

### 4.5 Experimental Results

We compare our 4 variants with StackGAN baseline method on CUB and Oxford datasets. In general, our variants outperform StackGAN and show their advantages in different aspects. The better averaged results indicate our proposed model is able to produce higher quality samples and contain more complete info.

As shown in Figure 3, the  $64 \times 64$  images are generated by StackGAN and our InfoGAN variants. We choose bird examples in different categories, colors and postures. Each line of figures are generated by the same text description listed in the first column. The first column of figures are ground-truth. The second column of figures are the baseline StackGAN results. And the following four columns are our four variants' results. We will first analyze them in different primitive properties like shape and color of the birds, and then compare the advantages and disadvantages of our four variants in details.











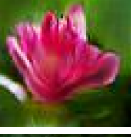
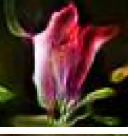




Description	Ground truth	Stack GAN	StageInfo_dfake_embed	StageInfo_dfake_dreal_embed
this flower has a brown center surrounded by layers of long yellow petals with rounded tips.				
this flower has long white petals and a large yellow stigma in the middle				
this flower is pink and white in color, with petals that are pointed upward.				
the flower is big and white with soft, smooth and big petals that has stamen in the center.				

Figure 4: Example results on Oxford Dataset by StackGAN StageI and our improved methods, i.e. StageInfo\_dfake\_embedded and StageInfo\_dfake\_dreal\_embed.

**Qualitative comparison:** First, let’s take a look at the first example figures in the first row of Figure 3. The text description is to generate a bird that is almost dark gray with long skinny wings and white head. For color property, the key words are “dark grey” and “white”. We can see that most of the results have correct dark grey color but only *StageInfo* presents the “white head” obviously. This shows that our Captioner component successfully completes more info in the generated result. As for the shape property, the key words are “skinny wings”. The baseline figure is even hard for us to distinguish the wings and also no “skinny” feature presented. In our InfoStack variants, we can see the shape of flying birds and *StageInfo* outperforms best. It also expressed the skinny shape property.

Let’s take a further look to the second row samples. This time we need to generate a brown ‘duck-like’ bird with hooked bill, long neck and white crown. Because this ‘duck-like’ shape is not common in the training set, the results are not as good as others. But we still shows the improvements from our models. We can see StackGAN and our *StageInfo* methods are fail to present any shape of birds. For our later variants, we can clearly see the brown feathers in *StageInfo\_dfake* method result, and white colors in all the four variants, and the long neck in *StageInfo\_dfake* and *StageInfo\_dfake\_dreal\_embed* methods.

This is a relatively difficult case, let’s see some easier examples. This time we focus on one primitive property, shape, in the third and fourth examples. The third row aims to draw a back bird with long tail and large curved beak. We can see the third variant *StageInfo\_dfake\_embed* method gives the best fit result. Long downward tail and large curved beak, which is exactly the same as the ground truth figure. Furthermore, the last row is an example to draw a bird that is completely back which has tiny brown bill and skinny body. Except for the posture and shape, this time we have another new property “skinny body”, which is also better expressed in the last four results.

The above analysis are based on the CUB dataset for birds. However, we didn’t see big advantages in the Oxford dataset for flowers. Because flowers have less variance in shapes and poses, the StackGAN method has already been able to capture enough information. The added Captioner component may be helpful but it’s not obvious.

**Component analysis:** We just list some typical results above to demonstrate the effectiveness of our approaches. Here we can analyze different variants of InfoStack on CUB bird dataset with baseline StackGAN. Generally speaking, compared to StackGAN, the *StageInfo* method can maintain almost all main color features. The shape feature gives either extremely good or terribly bad

	CUB		Oxford		mean
	realistic	relevance	realistic	relevance	
StackGAN	1.91	1.78	1.62	<b>2.04</b>	1.84
StageInfo_dfake_embed	<b>2.07</b>	<b>2.15</b>	<b>2.27</b>	1.95	<b>2.11</b>
StageInfo_dfake_dreal_embed	2.02	2.05	2.11	2.02	2.05

Table 1: Average Human Rating

results. So this indicates that our Captioner design does help maintain main more information of text descriptions to image, but it’s hard to capture the true relationship among the features. Furthermore, adding fake image info into our model reduces the failure rate and makes the generated figure more robust. The either extremely good or terribly bad issue does not happen in StageInfo\_dfake results. One more improvement is that, we find that taking use of embedding info rather than the real image data gives better performance. No matter for the color, shape, smoothness or even image resolution, the resulted figure of StageInfo\_dfake\_embed have better quality. Finally, we also add real image info, which seems not to help much in primitive properties. But we find that normally the results from StageInfo\_dfake\_dreal\_embed method present the similar contexts with the ground-truth figure, like birds flying in the sky, sitting on a stick, swimming in a lake, etc.

All in all, we can say that generally speaking, our proposed InfoStack approaches outperform StackGAN in primitive properties comparison. But among the four variants, we cannot say which one is the best. Each variants has its pros and cons and different improvements expressed on different samples.

**User Study:** To better present a objecti arison on the visual results, we conduct a user study by questionnaire survey. In the questionnaire, the users are asked to rank images generated from 3 methods, StackGan, StageInfo\_dfake\_embed and StageInfo\_dfake\_dreal\_embed, by their degree of realistic without text descriptions or by their degree of relevance to the given text description. For both CUB dataset and Oxford dataset, we randomly sample 10 groups of images with the same text description for each group. During the survey, the identifier of each image is hidden and the image orders within each group are randomly shuffled. When ranking 1st, 2nd or 3rd, 3, 2 or 1 credits will be assigned respectively. The results of average human rating are shown in Tab. 1.

From the user study, we can see that in the CUB dataset, the StageInfo\_dfake\_embed method and the StageInfo\_dfake\_dreal\_embed method have similar performance and they both outperform the StackGAN method in terms of realistic and relevance. In the Oxford dataset, the StageInfo\_dfake\_embed method and the StageInfo\_dfake\_dreal\_embed method outperform the StackGAN method in degree of realistic, but not in degree of relevance. Overall we can see that our proposed methods with information flow can generate more visually plausible and more relevant results than the baseline method StackGAN without information flow.

## 5 Conclusions

In this paper, we proposed a new neural network with information flow which adds a third component ‘Captioner’ to GAN structor. The key idea is to utilize the recovered original text from the generated image and feed it back to our network architecture, which is better expressed in the chain manner: text  $\rightarrow$  image  $\rightarrow$  text. Based on the existing framework StackGAN, we compare it with our four variant implementations. Our experiments show that the Captioner component can keep more primitive features and express more genuine info in the generated images. Furthermore, taking the embeddings in error calculation helps to improve the image info quality. In addition, generated fake image info and real image info are also helpful to improve the reliability of quality. Our work is promising to be further improved by combining these variants of implements and show its advantage in more advanced GAN architectures with more sophisticated datasets.

## Acknowledgments

This work is our final project in course CS388 NLP, UT Austin, Spring 2018. We are grateful to Prof. Raymond J. Mooney and TA Ghufra Baig for their insightful comments and help.



## References

- Martín Arjovsky and Léon Bottou. 2017. Towards principled methods for training generative adversarial networks. *CoRR*, abs/1701.04862.
- Xinyuan Chen, Chang Xu, Xiaokang Yang, and Dacheng Tao. 2018. Attention-gan for object transfiguration in wild images. *CoRR*, abs/1803.06798.
- Ronan Collobert and Ronan Collobert. 2015. Phrase-based image captioning. pages 2085–2094.
- Emily L. Denton, Soumith Chintala, Arthur Szlam, and Rob Fergus. 2015. Deep generative image models using a laplacian pyramid of adversarial networks. In *NIPS*.
- Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680.
- Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. 2015. Draw: A recurrent neural network for image generation. In *ICML*.
- A Karpathy and L Fei-Fei. 2017. Deep visual-semantic alignments for generating image descriptions. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, 39(4):664–676.
- Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.
- Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational bayes. *CoRR*, abs/1312.6114.
- Elman Mansimov, Emilio Parisotto, Jimmy Ba, and Ruslan Salakhutdinov. 2015. Generating images from captions with attention. *CoRR*, abs/1511.02793.
- Mehdi Mirza and Simon Osindero. 2014. Conditional generative adversarial nets. *CoRR*, abs/1411.1784.
- Maria-Elena Nilsback and Andrew Zisserman. 2008. Automated flower classification over a large number of classes. *2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing*, pages 722–729.
- Scott E. Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. 2016a. Learning deep representations of fine-grained visual descriptions. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 49–58.
- Scott E. Reed, Zeynep Akata, Xinchun Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. 2016b. Generative adversarial text to image synthesis. In *ICML*.
- Tim Salimans, Ian J. Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved techniques for training gans. In *NIPS*.
- C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. 2011. The Caltech-UCSD Birds-200-2011 Dataset. Technical Report CNS-TR-2011-001, California Institute of Technology.
- Jianwei Yang, Anitha Kannan, Dhruv Batra, and Devi Parikh. 2017. Lr-gan: Layered recursive generative adversarial networks for image generation. *CoRR*, abs/1703.01560.
- Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaolei Huang, Xiaogang Wang, and Dimitris Metaxas. 2017a. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *IEEE Int. Conf. Comput. Vision (ICCV)*, pages 5907–5915.
- Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas. 2017b. Stackgan++: Realistic image synthesis with stacked generative adversarial networks. *CoRR*, abs/1710.10916.