DM-GAN: Dynamic Memory Generative Adversarial Networks for Text-to-image Synthesis

Minfeng Zhu\textsuperscript{1,3\ast} Pingbo Pan\textsuperscript{3} Wei Chen\textsuperscript{1} Yi Yang\textsuperscript{2,3†}
\textsuperscript{1} State Key Lab of CAD&CG, Zhejiang University \textsuperscript{2} Baidu Research \textsuperscript{3} Centre for Artificial Intelligence, University of Technology Sydney
\{minfeng.zhu@, chenwei@cad\}zju.edu.cn \{pingbo.pan@student, Yi.Yang@\}uts.edu.au

Abstract

In this paper, we focus on generating realistic images from text descriptions. Current methods first generate an initial image with rough shape and color, and then refine the initial image to a high-resolution one. Most existing text-to-image synthesis methods have two main problems. (1) These methods depend heavily on the quality of the initial images. If the initial image is not well initialized, the following processes can hardly refine the image to a satisfactory quality. (2) Each word contributes a different level of importance when depicting different image content, however, unchanged text representation is used in existing image refinement processes. In this paper, we propose the Dynamic Memory Generative Adversarial Network (DM-GAN) to generate high-quality images. The proposed method introduces a dynamic memory module to refine fuzzy image contents, when the initial images are not well generated. A memory writing gate is designed to select the important text information based on the initial image content, which enables our method to accurately generate images from the text description. We also utilize a response gate to adaptively fuse the information read from the memories and the image features. We evaluate the DM-GAN model on the Caltech-UCSD Birds 200 dataset and the Microsoft Common Objects in Context dataset. Experimental results demonstrate that our DM-GAN model performs favorably against the state-of-the-art approaches.

1. Introduction

The last few years have seen remarkable growth in the use of Generative Adversarial Networks (GANs) \[4\] for image and video generation. Recently, GANs have been widely used to generate photo-realistic images according to certain text descriptions (see Figure 1). Fully understanding the relationship between visual contents and natural languages is an essential step towards artificial intelligence, e.g., image search and video understanding \[33\]. Multi-stage methods \[28\, 30\, 31\] first generate low-resolution initial images and then refine the initial images to high-resolution ones.

Although these multi-stage methods achieve remarkable progress, there remain two problems. First, the generation result depends heavily on the quality of the initial image. The image refinement process cannot generate high-quality images, if the initial images are badly generated. Second, each word in an input sentence has a different level of information depicting the image content. Current models utilize the same word representations in different image refinement processes, which makes the refinement process ineffective. The image information should be taken into account to determine the importance of every word for refinement.

In this paper, we introduce a novel Dynamic Memory Generative Adversarial Network (DM-GAN) to address the aforementioned issues. For the first issue, we propose to add a memory mechanism to cope with badly-generated initial images. Recent work \[27\] has shown the memory net-
work’s ability to encode knowledge sources. Inspired by this work, we propose to add the key-value memory structure \cite{13} to the GAN framework. The fuzzy image features of initial images are treated as queries to read features from the memory module. The reads of the memory are used to refine the initial images. To solve the second issue, we introduce a memory writing gate to dynamically select the words that are relevant to the generated image. This makes our generated image well conditioned on the text description. Therefore, the memory component is written and read dynamically at each image refinement process according to the initial image and text information. In addition, instead of directly concatenating image and memory, a response gate is used to adaptively receive information from image and memory.

We conducted experiments to evaluate the DM-GAN model on the Caltech-UCSD Birds 200 (CUB) dataset and the Microsoft Common Objects in Context (COCO) dataset. The quality of generated images is measured using the Inception Score (IS), the Fréchet Inception Distance (FID) and the R-precision. The experiments illustrate that our DM-GAN model outperforms the previous image-to-text synthesis methods, quantitatively and qualitatively. Our model improves the IS from 4.36 to 4.75 and decreases the FID from 23.98 to 16.09 on the CUB dataset. The R-precision is improved by 4.49% and 3.09% on the above two datasets. The qualitative evaluation proves that our model generates more photo-realistic images.

This paper makes the following key contributions:

- We propose a novel GAN model combined with a dynamic memory component to generate high-quality images even if the initial image is not well generated.
- We introduce a memory writing gate that is capable of selecting relevant word according to the initial image.
- A response gate is proposed to adaptively fuse information from image and memory.
- The experimental results demonstrate that the DM-GAN outperforms the state-of-the-art approaches.

2. Related Work

2.1. Generative Adversarial Networks.

With the recent successes of Variational Autoencoders (VAEs) \cite{9} and GANs \cite{4}, a large number of methods have been proposed to handle generation \cite{14,17,28,1} and domain adaptation task \cite{25,32}. Recently, generating images based on the text descriptions gains interest in the research community nowadays.

Single-stage. The text-to-image synthesis problem is decomposed by Reed \textit{et al.} \cite{20} into two sub-problems: first, the joint embedding is learned to capture the relations between natural language and real-world images; second, a deep convolutional generative adversarial network \cite{19} is trained to synthesize a compelling image. \textit{Dong et al.} \cite{3} adopted the pair-wise ranking loss \cite{10} to project both images and natural languages into a joint embedding space. Since previous generative models failed to add the location information, \textit{Reed et al.} proposed GAWWN \cite{21} to encode localization constraints. To diversify the generated images, the discriminator of TAC-GAN \cite{2} not only distinguishes real images from synthetic images, but also classifies synthetic images into true classes. Similar to TAC-GAN, PPGN \cite{16} includes a conditional network to synthesize images conditioned on a caption.

Multi-stage. StackGAN \cite{30} and StackGAN++ \cite{31} generate photo-realistic high-resolution images with two stages. \textit{Yuan et al.} \cite{29} employed symmetrical distillation networks to minimize the multi-level difference between real and synthetic images. DA-GAN \cite{12} translates each word into a sub-region of an image. Our method considers the interaction between each word and the whole generated image. Conditioning on the global sentence vector may result in low-quality images, AttnGAN \cite{28} refines the images to high-resolution ones by leveraging the attention mechanism. Each word in an input sentence has a different level of information depicting the image content. However, AttnGAN takes all the words equally, it employs an attention module to use the same word representation. Our proposed memory module is able to uncover such difference for image generation, as it dynamically selects the important word information based on the initial image content.

2.2. Memory Networks.

Recently, memory network \cite{5,27} provides a new architecture to reason answers from memories more effectively using explicit storage and a notion of attention. Memory network first writes information into an external memory and then reads contents from memory slots according to a relevance probability. \textit{Weston et al.} \cite{27} introduced the memory network to produce the output by searching supporting memories one by one. End-to-end memory network \cite{23} is a continues form of memory network, where each memory slot is weighted according to the inner product between the memory and the query. To understand the unstructured documents, the Key-Value Memory Network (KV-MemNN) \cite{13} performs reasoning by utilizing different encodings for key memory and value memory. The key memory is used to infer the weight of the corresponding value memory when predicting the final answer. Inspired by the recent success of the memory network, we introduce DM-GAN, a novel network architecture to generate high-quality images via nontrivial transforms between key and value memories.
3. DM-GAN

As shown in Figure 2, the architecture of our DM-GAN model is composed of two stages: initial image generation and dynamic memory based image refinement.

At the initial image generation stage, firstly, the input text description is transformed into some internal representation (a sentence feature $s$ and several word features $W$) by a text encoder. Then, a deep conventional generator predicts an initial image $x_0$ with a rough shape and few details according to the sentence feature and a random noise vector $z$: $x_0, R_0 = G_0(z, s)$, where $R_0$ is the image feature. The noise vector is sampled from a normal distribution.

At the dynamic memory based image refinement stage, more fine-grained visual contents are added to the fuzzy initial images to generate a photo-realistic image $x_i$: $x_i = G_i(R_{i-1}, W)$, where $R_{i-1}$ is the image feature from the last stage. The refinement stage can be repeated multiple times to retrieve more pertinent information and generate a high-resolution image with more fine-grained details.

The dynamic memory based image refinement stage consists of four components: Memory Writing, Key Addressing, Value Reading, and Response (Section 3.1). The Memory Writing operation stores the text information into a key-value structured memory for further retrieval. Then, Key Addressing and Value Reading operations are employed to read features from the memory module to refine the visual features of the low-quality images. At last, the Response operation is adopted to control the fusion of the image features and the reads of the memory. We propose a memory writing gate to highlight important word information according to the image content in memory writing step (Section 3.2).

We also utilize a response gate to adaptively fuse the information read from the memory and the image features (Section 3.3).

3.1. Dynamic Memory

We start with the given input word representations $W$, image $x$ and image features $R_i$:

$$W = \{w_1, w_2, ..., w_T\}, w_i \in \mathbb{R}^{N_w},$$
$$R_i = \{r_1, r_2, ..., r_N\}, r_i \in \mathbb{R}^{N_i},$$

where $T$ is the number of words, $N_w$ is the dimension of word features, $N$ is the number of image pixels and image pixel feature is a $N_r$ dimensional vector. We are intended to learn a model to refine the image using a more effective way to fuse text and image information via nontrivial transforms between key and value memory. The refinement stage includes the following four steps.

**Memory Writing**: Encoding prior knowledge is an important part of the dynamic memory, which enables recovering high-quality images from text. A naive way to write the memory is considering only partial text information. We propose a memory writing gate to highlight important word information according to the image content in memory writing step (Section 3.2).

$$m_i = M(w_i), m_i \in \mathbb{R}^{N_m}$$

where $M(\cdot)$ denotes the $1 \times 1$ convolution operation which embeds word features into the memory feature space with $N_m$ dimensions.

**Key Addressing**: In this step, we retrieve relevant memories using key memory. We compute a weight of each memory slot as a similarity probability between a memory
slot \( m_i \) and an image feature \( r_j \):
\[
\alpha_{i,j} = \frac{\exp(\phi_K(m_i)T r_j)}{\sum_{i=1}^{T} \exp(\phi_K(m_i)T r_j)},
\]
where \( \alpha_{i,j} \) is the similarity probability between the \( i \)-th memory and the \( j \)-th image feature and \( \phi_K() \) is the key memory access process which maps memory features into dimension \( N_r \). \( \phi_K() \) is implemented as a 1\( \times \)1 convolution.

**Value Reading:** The output memory representation is defined as the weighted summation of value memories according to the similarity probability:
\[
o_j = \sum_{i=1}^{T} \alpha_{i,j} \phi_V(m_i),
\]
where \( \phi_V() \) is the value memory access process which maps memory features into dimension \( N_v \). \( \phi_V() \) is implemented as a 1\( \times \)1 convolution.

**Response:** After receiving the output memory, we combine the current image and the output representation to provide a new image feature. A naive approach will be simply concatenating the image features and the output representation. The new image features are obtained by:
\[
r^{\text{new}}_i = [o_i, r_i],
\]
where \([\cdot, \cdot] \) denotes concatenation operation. Then, we are able to utilize an upsampling block and several residual blocks \([6]\) to upscale the new image features into a high-resolution image. The upsampling block consists of a nearest neighbor upsampling layer and a 3\( \times \)3 convolution. Finally, the refined image \( x \) is obtained from the new image features using a 3\( \times \)3 convolution.

### 3.2. Gated Memory Writing

Instead of considering only partial text information using Eq[3] the memory writing gate allows the DM-GAN model to select the relevant word to refine the initial images. The memory writing gate \( g^w \) combines image features \( R_i \) from the last stage with word features \( W \) to calculate the importance of a word:
\[
g^w_i(R_i, w_i) = A \ast w_i + B \ast \frac{1}{N} \sum_{i=1}^{N} r_i,
\]
where \( A \) is a \( 1 \times N_w \) matrix and \( B \) is a \( 1 \times N_r \) matrix. Then, the memory slot \( m_i \in R^{N_m} \) is written by combining the image and word features.
\[
m_i = M_w(w_i) \ast g_i^w + M_r(\frac{1}{N} \sum_{i=1}^{N} r_i) \ast (1 - g_i^w),
\]
where \( M_w() \) and \( M_r() \) denote the 1x1 convolution operation. \( M_w() \) and \( M_r() \) embed image and word features into the same feature space with \( N_m \) dimensions.

### 3.3. Gated Response

We utilize the adaptive gating mechanism to dynamically control the information flow and update image features:
\[
g^r_i = \sigma(W[o_i, r_i] + b),
\]
\[
r_i^{\text{new}} = o_i \ast g^r_i + r_i \ast (1 - g^r_i),
\]
where \( g^r_i \) is the response gate for information fusion, \( \sigma \) is the sigmoid function, \( W \) and \( b \) are the parameter matrix and bias term.

### 3.4. Objective Function

The objective function of the generator network is defined as:
\[
L = \sum_i L_{G_i} + \lambda_1 L_{CA} + \lambda_2 L_{DAMSM},
\]
in which \( \lambda_1 \) and \( \lambda_2 \) are the corresponding weights of conditioning augmentation loss and DAMSM loss. \( G_0 \) denotes the generator of the initial generation stage. \( G_i \) denotes the generator of the \( i \)-th iteration of the image refinement stage. The adversarial loss for each discriminator \( D_i \) is defined as follows:
\[
L_{D_i} = -\frac{1}{2}[E_{x \sim p_{data}} \log D_i(x) + E_{x \sim p_{G_i}} \log (1 - D_i(x))],
\]
where the first term is the unconditional loss which makes the generated image real as much as possible and the second term is the conditional loss which makes the image match the input sentence. Alternatively, the adversarial loss for each discriminator \( D_i \) is defined as:
\[
L_{D_i} = -\frac{1}{2}[E_{x \sim p_{data}} \log D_i(x) + E_{x \sim p_{G_i}} \log (1 - D_i(x)) + E_{x \sim p_{data}} \log (1 - D_i(x, s))]
\]
\[
\quad + E_{x \sim p_{G_i}} \log (1 - D_i(x, s))],
\]
where the unconditional loss is designed to distinguish the generated image from real images and the conditional loss determines whether the image and the input sentence match.

**Conditioning Augmentation Loss:** The Conditioning Augmentation (CA) technique \([30]\) is proposed to augment training data and avoid overfitting by resampling the input sentence vector from an independent Gaussian distribution. Thus, the CA loss is defined as the Kullback-Leibler divergence between the standard Gaussian distribution and the Gaussian distribution of training data.
\[
L_{CA} = D_{KL}(\mathcal{N}(\mu(s), \Sigma(s)) || \mathcal{N}(0, I)),
\]
where \( \mu(s) \) and \( \Sigma(s) \) are mean and diagonal covariance matrix of the sentence feature. \( \mu(s) \) and \( \Sigma(s) \) are computed by fully connected layers.
**DAMSM Loss**: We utilize the DAMSM loss [28] to measure the matching degree between images and text descriptions. The DAMSM loss makes generated images better conditioned on text descriptions.

### 3.5. Implementation Details

For text embedding, we employ a pre-trained bidirectional LSTM text encoder by Xu et al. [28] and fix their parameters during training. Each word feature corresponds to the hidden states of two directions. The sentence feature is generated by concatenating the last hidden states of two directions. The initial image generation stage first synthesizes images with 64x64 resolution. Then, the dynamic memory based image refinement stage refines images to 128x128 and 256x256 resolution. We only repeat the refinement process with dynamic memory module two times due to GPU memory limitation. Introducing dynamic memory to low-resolution images (i.e., 16x16, 32x32) cannot improve the performance. Because low-resolution images are not well generated and their features are more like random vectors. For all discriminator networks, we apply spectral normalization [15] after every convolution to avoid unusual gradients to improve text-to-image synthesis performance. By default, we set $N_w = 256$, $N_r = 64$ and $N_m = 128$ to be the dimension of text, image and memory feature vectors respectively. We set the hyperparameter $\lambda_1 = 1$ and $\lambda_2 = 5$ for the CUB dataset and $\lambda_1 = 1$ and $\lambda_2 = 50$ for the COCO dataset. All networks are trained using ADAM optimizer [8] with batch size 10, $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The learning rate is set to be 0.0002. We train the DM-GAN model with 600 epochs on the CUB dataset and 120 epochs on the COCO dataset.

### 4. Experiments

In this section, we evaluate the DM-GAN model quantitatively and qualitatively. We implemented the DM-GAN model using the open-source Python library PyTorch [18].

**Datasets.** To demonstrate the capability of our proposed method for text-to-image synthesis, we conducted experiments on the CUB [26] and the COCO [11] datasets. The CUB dataset contains 200 bird categories with 11,788 images, where 150 categories with 8,855 images are employed for training while the remaining 50 categories with 2,933 images for testing. There are ten captions for each image in CUB dataset. The COCO dataset includes a training set with 80k images and a test set with 40k images. Each image in the COCO dataset has five text descriptions.

**Evaluation Metric.** We quantify the performance of the DM-GAN in terms of Inception Score (IS), Fréchet Inception Distance (FID), and R-precision. Each model generated 30,000 images conditioning on the text descriptions from the unseen test set for evaluation.

The IS uses a pre-trained Inception v3 network [24] to compute the KL-divergence between the conditional class distribution and the marginal class distribution. A large IS means that the generated model outputs a high diversity of images for all classes and each image clearly belongs to a specific class.

The FID computes the Fréchet distance between synthetic and real-world images based on the extracted features from a pre-trained Inception v3 network. A lower FID implies a closer distance between generated image distribution and real-world image distribution.

Following Xu et al. [28], we use the R-precision to evaluate whether a generated image is well conditioned on the given text description. The R-precision is measured by retrieving relevant text given an image query. We compute the cosine distance between a global image vector and 100 candidate sentence vectors. The candidate text descriptions include R ground truth and 100-R randomly selected mismatching descriptions. For each query, if $r$ results in the top R ranked retrieval descriptions are relevant, then the R-precision is $r/R$. In practice, we compute the R-precision with $R=1$. We divide the generated images into ten folds for retrieval and then take the mean and standard deviation of the resulting scores.

### 4.1. Text-to-image Quality

We compare our DM-GAN model with the state-of-the-art models on the CUB and COCO test datasets. The performance results are reported in Table 1 and 2.

Table 1 compares the performance between AttnGAN and DM-GAN with respect to the FID on the CUB and COCO datasets. We measure the FID of AttnGAN from the officially pre-trained model. Our DM-GAN decreases the FID from 23.98 to 16.09 on the CUB dataset and from 35.49 to 32.64 on the COCO dataset, which demonstrates that DM-GAN learns a better data distribution.

As shown in Table 2, the DM-GAN improves the R-precision by 4.49% on the CUB dataset and 3.09% on the COCO dataset. Higher R-precision indicates that the generated images by the DM-GAN are better conditioned on the given text description, which further demonstrates the effectiveness of the employed dynamic memory.

In summary, the experimental results indicate that our DM-GAN is superior to the state-of-the-art models.
4.2. Visual Quality

For qualitative evaluation, Figure 3 shows text-to-image synthesis examples generated by our DM-GAN and the state-of-the-art models. In general, our DM-GAN approach generates images with more vivid details as well as more clear background in most cases, comparing to the AttnGAN [28], GAN-INT-CLS [20], GAWWN [21], StackGAN [30], PPGN [16], AttnGAN [28] and our DM-GAN on the CUB and COCO datasets. The best results are in bold.

Table 1. The inception scores (higher is better) of GAN-INT-CLS [20], GAWWN [21], StackGAN [30], PPGN [16], AttnGAN [28] and our DM-GAN on the CUB and COCO datasets. The best results are in bold.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CUB</td>
<td>FID↓</td>
<td>2.88±0.04</td>
<td>3.62±0.07</td>
<td>3.70±0.04</td>
<td>(-)</td>
<td>4.36±0.03</td>
<td>4.75±0.07</td>
</tr>
<tr>
<td>COCO</td>
<td>FID↓</td>
<td>7.88±0.07</td>
<td>(-)</td>
<td>8.45±0.03</td>
<td>9.58±0.21</td>
<td>25.89±0.47</td>
<td>30.49±0.57</td>
</tr>
</tbody>
</table>

Table 2. Performance of FID and R-precision for AttnGAN [28] and our DM-GAN on the CUB and COCO datasets. The FID of AttnGAN is calculated from officially released weights. Lower is better for FID and higher is better for R-precision.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Metric</th>
<th>AttnGAN</th>
<th>DM-GAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUB</td>
<td>FID↓</td>
<td>23.98</td>
<td>16.09</td>
</tr>
<tr>
<td></td>
<td>R-precision↑</td>
<td>67.82±4.43</td>
<td>72.31±0.91</td>
</tr>
<tr>
<td>COCO</td>
<td>FID↓</td>
<td>35.49</td>
<td>32.64</td>
</tr>
<tr>
<td></td>
<td>R-precision↑</td>
<td>85.47±3.69</td>
<td>88.56±0.28</td>
</tr>
</tbody>
</table>

Table 3. The performance of different architectures of our DM-GAN on the CUB datasets. M, WG and RG denote dynamic memory writing gate and response gate respectively.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>IS↑</th>
<th>FID↓</th>
<th>R-Precision↑</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>4.51±0.04</td>
<td>23.32</td>
<td>68.60±0.73</td>
</tr>
<tr>
<td>+M</td>
<td>4.57±0.05</td>
<td>21.41</td>
<td>70.66±0.69</td>
</tr>
<tr>
<td>+M+WG</td>
<td>4.65±0.05</td>
<td>20.83</td>
<td>71.40±0.64</td>
</tr>
<tr>
<td>+M+WG+RG</td>
<td>4.75±0.07</td>
<td>16.09</td>
<td>72.31±0.91</td>
</tr>
</tbody>
</table>

4.3. Ablation Study

In order to verify the effectiveness of our proposed components, we evaluate the DM-GAN architecture and its variants on the CUB dataset. The control components between architectures include the key-value memory (M), the writing gate (WG) and the response gate (RG). We define a baseline model which removes M, WG and RG from DM-GAN. The memory is written according to partial text information (Eq.3). The response operation simply concatenates the image features and the memory output (Eq.6). The performance of the DM-GAN architecture and its variants is reported in Table 3. Our baseline model produces slightly better performance than AttnGAN. By integrating these components, our model can achieve further improvement which

realistic high-resolution images. So the image quality is obviously well-improved, with more clear background and convincing details. In most cases, the initial stage generates a blurry image with rough shape and color, so that the background is fine-tuned to be more realistic with fine-grained textures, while the refined image will be better conditioned on the input text and provide more photo-realistic high-resolution images. In the fourth column of Figure 4, no white streaks can be found on the bird body from the initial image with 64×64 resolution. The refinement process helps to encode "white streaks" information from text description and add back missing features based on the text description and image content. In order word, our DM-GAN model is able to refine the image to match the input text description.

To evaluate the diversity of our DM-GAN model, we generate several images using the same text description, and multiple noise vectors. Figure 5 shows text descriptions and synthetic images with different shapes and backgrounds. Images are similar but not identical to each other, which means our DM-GAN generates images with high diversity.
<table>
<thead>
<tr>
<th>DM-GAN</th>
<th>AttnGAN</th>
<th>StackGAN</th>
<th>GAN-INT-CLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>This bird has wings that are grey and has a white belly.</td>
<td>This bird has wings that are black and has a white belly.</td>
<td>This is a grey bird with a brown wing and a small orange beak.</td>
<td>This bird has a short brown bill, a white eyering, and a medium brown crown.</td>
</tr>
<tr>
<td>This particular bird has a belly that is yellow and brown.</td>
<td>This bird is a lime green with greyish wings and long legs.</td>
<td>This yellow bird has a thin beak and jet black eyes and thin feet.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Room with wood floors and a stone fireplace.</th>
<th>The bathroom with the white tile has been cleaned.</th>
<th>A fruit stand that has bananas, papayas, and plantains.</th>
<th>A stop sign that is sitting in the grass.</th>
</tr>
</thead>
<tbody>
<tr>
<td>A train accident where some cars went into a river.</td>
<td>A bunch of various vegetables on a table.</td>
<td>A plane parked at an airport near a terminal.</td>
<td>A silhouette of a man surfing over waves.</td>
</tr>
</tbody>
</table>

Figure 3. Example results for text-to-image synthesis by DM-GAN and AttnGAN. (a) Generated bird images by conditioning on text from CUB test set. (b) Generated images by conditioning on text from COCO test set.
This bird has a blue crown with white throat and brown secondaries.

This small bird has a yellow crown and a white belly.

A primarily black bird with streaks of white and yellow and a medium sized beak.

People at the park flying kites and walking.

The bathroom with the white tile has been cleaned.

Multiple people are standing on the beach at the edge of the water.

A clock that is on the side of a tower.

This bird has wings that are grey and has a white belly.

A group of people standing on a beach next to the ocean.

This bird is red in color with a black and white breast and a black eyering.

(a) This bird is blue with white and has a very short beak.

(b) This bird is blue with white and has a very short beak.

Figure 4. The results of different stages of our DM-GAN model, including the initial images, the images after one refinement process and the images after two refinement processes.

Figure 5. Generated images using the same text description.

demonstrates the effectiveness of every component.

Further, we visualize the most relevant words selected by the AttnGAN [28] and our DM-GAN. We notice that the attention mechanism cannot accurately select relevant words when the initial images are not well generated. We propose the dynamic memory module to select the most relevant words based on the global image feature. As Fig. 6 (a) shows, although a bird with incorrect red breast is generated, dynamic memory module selects the word, i.e., "white" to correct the image. The DM-GAN selects and combines word information with image features in two steps (see Fig. 6(b)). The gated memory writing step first roughly selects words relevant to the image and writes them into the memory. Then the key addressing step further reads more relevant words from the memory.

5. Conclusions

In this paper, we have proposed a new architecture called DM-GAN for text-to-image synthesis task. We employ a dynamic memory component to refine the badly generated image, a memory writing gate to highlight important text information and a repose gate to fuse image and memory representation. Sometime the images synthesized by the DM-GAN are not satisfying. Because the complicated scenes of the COCO dataset make it challenging for most existing generative models, especially when more than one object needs to be generated. Our DA-GAN refines initial images with wrong color and rough shapes. However, the final results still rely heavily on the layout of multi-subjects in initial images. In the future, we will try to design a more powerful model to generate initial images with better layout.

Acknowledgment

This research has been supported by National Key Research and Development Program (2018YFB0904503) and National Natural Science Foundation of China (U1866602,61772456).
References


