Generative Adversarial Networks for Diverse and Explainable Text-to-Image Generation

Zhang, Zhenxing

DOI:
10.33612/diss.507581028

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version
Publisher's PDF, also known as Version of record

Publication date:
2023

Link to publication in University of Groningen/UMCG research database

Citation for published version (APA):
https://doi.org/10.33612/diss.507581028

Copyright
Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license. More information can be found on the University of Groningen website: https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment.

Take-down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): http://www.rug.nl/research/portal. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.
Most existing text-to-image generation methods adopt a multi-stage modular architecture which has three significant problems: (1) Training multiple networks increases the run time and affects the convergence and stability of a generative model; (2) These approaches ignore the quality of early-stage generator images; (3) Many discriminators need to be trained. To this end, we propose the Dual-Attention Generative-Adversarial Network (DTGAN) which can synthesize high-quality and semantically consistent images only employing a single generator/discriminator pair. The proposed model introduces channel-aware and pixel-aware attention modules that can guide the generator to focus on text-relevant channels and pixels based on the global sentence vector and to fine-tune original feature maps using attention weights. Also, Conditional Adaptive Instance-Layer Normalization (CAdaILN) is introduced to enable the linguistic cues from the sentence embedding to flexibly manipulate the amount of change in shape and texture, further improving visual-semantic representation and helping stabilize the training. Furthermore, a new type of visual loss is utilized to enhance the image resolution by ensuring vivid shape and perceptually uniform color distributions of generated images. Experimental results on benchmark data sets demonstrate the superiority of our proposed method compared to the state-of-the-art models with a multi-stage framework.

This chapter is based on
2.1 INTRODUCTION

The goal of text-to-image synthesis is to automatically yield perceptually plausible and semantically consistent pictures, given textual descriptions. Recently, this topic rapidly gained attention in computer vision (CV) and natural-language processing (NLP) research due to its extensive range of potential real-world applications including art creation, computer-aid design, data augmentation for training image classifiers, photo editing according to text descriptions, visually checking the output of captioning algorithms, et cetera. Nevertheless, text-to-image generation is still an extremely challenging cross-modal task, since it not only requires a thorough semantic understanding of natural-language descriptions, but also requires a conversion of textual-context features into high-resolution images. The common paradigm involves a deep generative model implementing the cross-domain information fusion.

Thanks to the recent advances in conditional generative-adversarial networks (cGANs) [71], current text-to-image approaches have made tremendous progress in image quality and semantic consistency when given natural-language descriptions at the inputs. Here, contrary to the plain GAN approach, conditional information can be injected into the generator network, modifying the manner in which a random latent code is transformed from input to output, e.g., using a text embedding or other external, conditional modifier. However, most text-to-image methods [55], [66], [83], [118], [123], [127], [140] are based on a multi-stage modular architecture, as shown in Fig. 2.1 (a). Specifically, a multi-stage framework is constructed with multiple generators and corresponding discriminators, producing visually realistic samples in a coarse-to-fine manner. Simultaneously, the output of an initial-stage network is fed into a next-stage network to generate an image with a higher resolution. This framework has proven to be useful for the task of text-to-image synthesis, but there still exist three significant problems. First, training many networks increases the computation time compared to a unified model and affects the convergence and stability of a generative model [105]. Even worse, the final generator network cannot be improved if the previous generators do not converge to a global optimum. Second, this architecture ignores the quality of early-stage generator images, which plays a vital role in the resolution of finally-generated images [140]. The generator networks for precursor images (G_0 in Fig. 2.1 (a)) are composed of up-sampling layers and convolution layers, lacking the image integration and refinement process with input natural-language descriptions. Third, multiple discriminators need to be trained.

To address the issues mentioned above, we propose a novel Dual-Attention Generative-Adversarial Network (DTGAN), which can fine-tune the feature maps for each scale according to given text descriptions and synthesize high-quality and semantically related images only using a single generator/discriminator
2.1 Introduction

The multi-stage framework (a) generates final images by training three generators and discriminators. The proposed DTGAN (b) is able to synthesize realistic images only using a single generator/discriminator pair. In (a), \( G_0 \)-\( G_2 \) are generators and \( D_0 \)-\( D_2 \) are discriminators. In (b), \( L_0 \)-\( L_6 \) are the dual-attention layers discussed in Section 2.4.2 and Section 2.4.3, and \( G \) and \( D \) are our generator and discriminator, respectively.

The overall architecture of DTGAN is illustrated in Fig. 2.1 (b). DTGAN consists of four new components, i.e., two new types of attention modules, a new normalization layer and a new type of visual loss. The first two components in DTGAN are our designed channel-aware and pixel-aware attention modules, which can guide the generator network to focus more on important channels and pixels and to ignore text-irrelevant channels and pixels by computing attention weights between the global sentence vector and two aforementioned factors. In contrast to earlier attention models [55], [118], we employ global average pooling and global max pooling to obtain the discriminative regions of image feature maps. In addition, we apply the attention scores to fine-tune original feature maps rather than adopt the weighted sum of converted word features as new feature maps. More importantly, we spread the attention over many, even all layers of the generator network. This allows for an influence of the text over features at various hierarchical levels in the pipeline architecture, from crude early features to abstract late features. We expect that our proposed attention methods will significantly bridge the semantic gap between generated images and text descriptions. In the third component, in order to stabilize the learning of a cGAN and boost the visual-semantic embedding in the visual feature map, Conditional Adaptive Instance-Layer Normalization (CAdaILN) is introduced to normalize the feature map in the layer and channel while employing the detailed and fine-grained linguistic cues to scale and shift the normalized feature map. Moreover, CAdaILN can help with flexibly controlling the amount of change in shape and texture, allowing for deeper networks and
complementing the modulation modules. For example, a good generator should be able to respond to the requirements specified by textual qualifiers for size (‘large’) and/or color (‘red’). As a result, armed with the presented attention modules and CAdaILN, our network can generate photo-realistic images only exploiting a single generator/discriminator pair. The last proposed component is a new variant for computing the visual loss. It is introduced to ensure that generated samples and real pictures have similar color distributions and shape. We expect that the choice of this novel visual loss has a considerable impact on the quality of synthesized images.

We perform extensive experiments on the CUB bird [108] and MSCOCO [63] data sets to evaluate the effectiveness of our proposed DTGAN. Both qualitative and quantitative results demonstrate that our approach outperforms existing state-of-the-art models. The contributions of our work can be summarized as follows:

- To the best of our knowledge, we are the first to present the fine-tuning on each scale of feature maps using the attention modules and the conditional normalization function, in order to generate high-quality and semantically consistent images only employing a single generator/discriminator pair.
- We design two new types of attention models to guide the generator to focus on text-relevant channels and pixels and spread the attention over almost all layers of the generator.
- CAdaILN is introduced to stabilize training as well as help modulation modules flexibly control the amount of change in shape and texture.
- We are the first to introduce the visual loss in text-to-image synthesis to enhance image quality.

2.2 RELATED WORK

In this section, we depict the research fields associated with our work, i.e., GANs, cGAN-based text-to-image generation and attention mechanism.

2.2.1 Generative adversarial networks (GANs)

Goodfellow et al. [30] presented the GAN paradigm that serves as a basic model for synthetic tasks via adversarial training and consists of a generator and a discriminator. A GAN has achieved state-of-the-art performance in a variety of applications, e.g., text-to-image synthesis [25], person image generation [96], face photo-sketch synthesis [119], image inpainting [129] and image de-raining [121], since it is capable of producing photo-realistic images.

The initial generator network of a GAN mainly comprises multi-layer perceptrons and rectifier linear activations, while the discriminator net utilizes the
maxout network [31]. This type of architecture shows competitive samples with other generative models on simple image data sets, such as MNIST [53]. Moreover, researchers explore different structures of a GAN in order to further improve image quality. Denton et al. [18] designed a Laplacian pyramid framework of an adversarial network namely LAPGAN that produces plausible pictures in a coarse-to-fine manner. Radford et al. [84] introduced a deep convolutional GAN (DCGAN) integrating convolutional layers and Batch Normalization (BN) into both a generator and a discriminator. Mirza et al. [71] proposed a cGAN by imposing conditional constraints (e.g., class labels, text descriptions and low-resolution images) on both the generator network and the discriminator network to obtain specific samples.

Recently, several models with a high-computational cost are introduced to yield visually plausible pictures. Zhang et al. [125] presented SAGAN which applies the self-attention mechanism [106] to effectively capture the semantic affinities between widely separated image regions. Brock et al. [9] developed a large-scale architecture based on SAGAN while deploying orthogonal regularization to the generator net, obtaining excellent performance on image diversity. Karras et al. [48] proposed a novel generator framework named StyleGAN where adaptive instance normalization is utilized to control the generator network. This chapter focuses on studying a conditional text-to-image GAN model.

2.2.2 cGAN in text-to-image generation

In recent years, the task of text-to-image synthesis has attracted rapidly growing attention from both CV and NLP communities. Thanks to the critical improvements in generative approaches, especially a GAN and a cGAN, inspiring advances in text-to-image generation have been made possible. According to the number of the generators and the discriminators these methods exploit, we roughly group them into two categories.

**Multi-stage.** Zhang et al. [126], [127] presented StackGAN and StackGAN++ to synthesize considerably compelling instances from textual descriptions in a coarse-to-fine way. Specifically, they utilized a multi-stage pipeline, in which each stage was composed of a generator and a discriminator, and the generator of the next stage received the result of the previous stage as the input. It is worth mentioning that StackGAN served as a solid basis for the future study of text-to-image synthesis. Qiao et al. [83] proposed MirrorGAN exploiting a semantic-preserving text-to-image-to-text framework, where an image caption model was customized to regenerate the text description from a generated sample in order to boost the semantic relevancy between textual contexts and visual content. Zhu et al. [140] developed DMGAN which applied a dynamic memory module to improve the quality of initial images. Yin et al. [122] designed a Siamese structure and conditional BN to implement semantic consistency of synthetic
images. Li et al. [58] built ObjGAN that paid attention to salient-object generation in complex scene. To be specific, ObjGAN leveraged a pre-generated layout to synthesize a multi-object image and employed an object-aware discriminator to judge whether generated objects match textual descriptions. Similar to ObjGAN, OP-GAN explicitly modeled the objects of an image, presented by Hinz et al. [39]. SegAttnGAN [32] proposed to employ additional segmentation information for the process of image refinement such that the network is capable of yielding images with realistic quality. CPGAN [61] designed a memory structure to parse the produced image in an object-wise manner and introduced a conditional discriminator to promote the semantic alignment of text-image pairs.

**Single-stage.** Reed et al. [84] were the first to use a single-stage cGAN to generate images from detailed text descriptions. However, the quality of synthesized images is limited due to the simple structure and immature training techniques of a cGAN. Tao et al. [105] presented DF-GAN adopting a matching-aware zero-centered gradient penalty (MA-GP) loss to cope with the problems of a complicated multi-stage text-to-image architecture. Nonetheless, DF-GAN just made use of a fully-connected layer to merge the feature map and a sentence vector, lacking an efficient modulation mechanism.

### 2.2.3 Attention mechanism

One vital property of our human visual system is that humans are capable of focusing more on the salient parts of an image and ignoring unimportant regions. Inspired by this, attention mechanisms are invented to guide the network to concentrate on the most discriminative local features and filter out irrelevant information. Thanks to their advantages over traditional methods with respect to feature processing and feature selection, attention mechanisms have been extensively explored in a series of CV fields, such as automatic segmentation [69], [104], image fusion [21], object tracking [124], video deblurring [130], scene classification [8], image super-resolution [111] and action recognition [136].

There have been attention ways in text-to-image synthesis, since attention mechanisms play an essential role in bridging the semantic gap between vision and language. On the one hand, Xu et al. [118] utilized a spatial-attention mechanism to derive the relationship between the image subregions and the words in a sentence. The most relevant subregions to the words were particularly focused. On the other hand, Li et al. [55] designed a channel-wise attention mechanism on the basis of Xu et al. [118]. However, the aforementioned works adopt the weighted sum of converted word features as the new feature map which is largely different from the original feature map. Moreover, they both equally treat all pixels and channels playing different roles in generating samples.

As opposed to them, we propose to spread the attention weights over all layers of the generator network. This allows for an influence of the text over features at
various hierarchical levels in the pipeline architecture, from crude early features to abstract late features. To improve image quality and stabilize the learning of a cGAN, we introduce the residual structure to retain the basic features to some extent. In the meantime, we perform global average pooling and global max pooling on feature maps to extract significant features. Experimental results conducted on the CUB bird and MSCOCO data sets show the superiority of our proposed attention modules compared to the aforementioned methods.

2.3 PRELIMINARIES

2.3.1 Generative adversarial networks (GANs)

As shown in Fig. 2.2, a GAN comprises two nets: the generator network $G$ and the discriminator network $D$, which are perceived as playing a minmax zero-sum game. To be concrete, the aim of $G$ is to capture the distribution of real data while yielding plausible images to trick $D$, whereas $D$ is optimized to classify a sample as real or fake. Simultaneously, $G$ and $D$ are typically implemented by deep neural networks. Mathematically, the minmax objective $V(G, D)$ for the GAN paradigm can be denoted as follows:

$$
\min_{G} \max_{D} V(G, D) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x)] + \mathbb{E}_{z \sim p_{z}(z)} [\log(1 − D(G(z)))]
$$

where $p_{data}(x)$ and $p_{z}(z)$ represent the distribution of true data $x$ and a random latent code $z$, respectively.

$G$ tries to minimize the objective $V$ during the minmax two-player game, while $D$ aims to maximize it. This minmax zero-sum game is finished when the distribution of produced samples entirely overlaps $p_{data}(x)$.

2.3.2 Conditional generative adversarial networks (cGANS)

As an extension of a GAN, a cGAN takes the conditional contexts $c$ (e.g. class labels, text descriptions and low-resolution images) and a random code $z$ as the inputs of $G$, while also outputting the samples which are semantically correlated with $c$. In the meantime, $D$ entails distinguishing the real image-context pair $(x, c)$ from the fake image-context pair $(G(z, c), c)$. Concretely, the objective $V(G, D)$ for a cGAN is formulated as:

$$
\min_{G} \max_{D} V(G, D) = \mathbb{E}_{x \sim p_{data}(x)} [\log D(x, c)] + \mathbb{E}_{z \sim p_{z}(z), c \sim p_{c}(c)} [\log(1 − D(G(z, c), c))]
$$
where $p_c(c)$ denotes the distribution of $c$.

In the task of text-to-image synthesis, $c$ is a given textual description. Let $\{(I_i, C_i)\}_{i=1}^n$ denote a set of $n$ image-text pairs for training, where $I_i$ represents an image and $C_i = (c_i^1, c_i^2, ..., c_i^k)$ indicates a suite of $k$ natural-language descriptions. The goal of $G$ is to produce a visually plausible and semantically consistent sample $\hat{I}_i$ according to a text description $c_i$ randomly picked from $C_i$, where $c_i = (w_1, w_2, ..., w_m)$ contains $m$ words. At the same time, $D$ is trained to distinguish the real image-text pair $(I_i, c_i)$ from the synthetic image-text pair $(\hat{I}_i, c_i)$.

2.3.3 Scaled dot-product attention

The fusion model involves bridging the semantic gap between different modalities through integrating the extracted features from different domains into one compact cross-modality representation. It can be easily joined to a pipeline without any need for the changes of a neural-network architecture. The attention module that is widely explored for fusion models enables a network to effectively learn the alignments between different modalities. Generally, an attention function \cite{vaswani2017attention} receives a query $q_i \in \mathbb{R}^d$ and a series of key-value pairs as the inputs and outputs a weighted sum of the values, where the attention weight on each value is calculated via a softmax function on the dot products of the query with all keys. The process of generating attention weights is formulated as follows:

$$
Attention(q_i, K, V) = \text{Softmax}\left(\frac{D(q_i, K^T)}{\sqrt{d}}\right)V
$$

(2.3)
The bird is a mixture of yellow, black, and white with a sharp pointed beak.

where \( D(\cdot) \) denotes the dot-product operation and \( \text{Softmax} \) represents the softmax function. \( K = (k_1, k_2, ..., k_n), k_i \in \mathbb{R}^d \), and \( V = (v_1, v_2, ..., v_n), v_i \in \mathbb{R}^d \), refer to the keys and the values, respectively. \( \frac{1}{\sqrt{d}} \) is the scaling factor.

2.4 PROPOSED METHODOLOGY

In this section, we elaborate on our proposed DTGAN which is shown in Fig. 2.3. Unlike prior works [55], [66], [83], [118], [123], [126], [127], [140], our goal is to generate a high-quality and visually realistic image which semantically aligns with a given natural-language description only employing a single generator/discriminator pair. To this end, we present four significant components: a channel-aware attention module, a pixel-aware attention module, Conditional Adaptive Instance-Layer Normalization (CAdaILN) and a new type of visual loss. Each of them will be discussed in detail after briefly describing the overall framework of DTGAN.
2.4.1 Overall architecture

As shown in Fig. 2.3, our architecture is composed of a text encoder and a generator/discriminator pair. For the text encoder, we adopt a bidirectional Long Short-Term Memory (LSTM) network \[92\] to learn the semantic representation of a given text description. Specifically, in the bidirectional LSTM layer, two hidden states are employed to capture the semantic meaning of words and the last hidden state is utilized to represent the sentence feature. The generator network of DTGAN takes a global sentence vector and a noise vector as the inputs and consists of seven dual-attention layers that are responsible for different scales of feature maps. Each dual-attention layer comprises two convolution layers, two CAdaILN layers, a channel-aware attention module and a pixel-aware attention module. Mathematically,

\[
h_0 = F_0(z) \\
h_1 = F_{1D}^\text{Dual}(h_0, s) \\
h_i = F_{iD}^\text{Dual}(h_{i-1} \uparrow, s) \quad \text{for} \quad i = 2, 3, ..., 7 \\
o = G_c(h_7)
\]

where \(z\) is a noise vector from the normal distribution, \(F_0\) is a fully-connected layer, \(F_{iD}^\text{Dual}\) is our proposed dual-attention layer, \(G_c\) is the last convolution layer, \(h_0\) is the output of the first fully-connected layer, \(h_1-h_7\) are the outputs of dual-attention layers and \(o\) is the generated image.

2.4.2 Channel-aware attention module

The feature map of each channel at the convolution layer plays different roles in generating the image which is semantically correlated with the given text description. Without fine-tuning the channel maps at the generative stage according the input natural-language description, the generated result may lack the semantic relevancy to the given textual description. Thus, we introduce a channel-aware attention module to guide the generator to focus on text-relevant channels and ignore minor channels.

The process of our channel-aware attention module is shown in Fig. 2.4. The channel-aware attention module takes two inputs: the feature map \(h\) and the global sentence vector \(s \in R^D\). Firstly, we perform global average pooling and global max pooling on \(h\) to extract the channel features: \(x_a \in R^{C \times 1 \times 1}\) and \(x_m \in R^{C \times 1 \times 1}\). Global average pooling is to obtain the information of the whole
2.4 Proposed Methodology

Figure 2.4: Overview of the proposed channel-aware attention module. GAP and GMP denote global average pooling and global max pooling, respectively.

feature map, while global max pooling focuses on the most discriminative part [137]. Mathematically,

\[
x_a = \text{GAP}(h) \\
x_m = \text{GMP}(h)
\]

(2.8) \hfill (2.9)

where GAP denotes global average pooling, GMP is global max pooling.

Then, we adopt a query, key and value setting [106] to capture the semantic relevancy between channels and the input text, where \( x_a \) and \( x_m \) are used as the query and \( s \) is selected as the key and the value. It is defined as:

\[
q_{ac} = W_{qa}x_a, q_{mc} = W_{qm}x_m \\
k_c = W_{kc}s, v_c = W_{vc} \]

(2.10) \hfill (2.11)

where \( W_{qa}, W_{qm}, W_{kc} \) and \( W_{vc} \) are the projection matrixes.

Assuming that the dot products [106] between the sentence-level key \( k_c^T \in \mathbb{R}^{1 \times D} \) and the average-pooling query \( q_{ac} \in \mathbb{R}^{C \times 1} \), the max-pooling query \( q_{mc} \in \mathbb{R}^{C \times 1} \) can capture meaningful features, the attention scores of channel maps are achieved through the following attention mechanism:

\[
\tilde{\alpha}_a^c = q_{ac} \cdot k_c^T, \tilde{\alpha}_m^c = q_{mc} \cdot k_c^T \\
\alpha_a^c = \text{softmax}(\tilde{\alpha}_a^c \cdot v_c) \\
\alpha_m^c = \text{softmax}(\tilde{\alpha}_m^c \cdot v_c)
\]

(2.12) \hfill (2.13) \hfill (2.14)
where ˜\(\alpha_c\) ∈ \(\mathbb{R}^{C \times D}\) and ˜\(\alpha_m\) ∈ \(\mathbb{R}^{C \times D}\) represent the semantic similarity between channel maps and the global sentence vector, \(\alpha^c\) ∈ \(\mathbb{R}^{C \times 1}\) and \(\alpha^m\) ∈ \(\mathbb{R}^{C \times 1}\) denote the final attention weights of channels for global average pooling and global max pooling, respectively, ˜\(\alpha_a\), ˜\(\alpha_m\), \(\alpha_a\) and \(\alpha_m\) are all computed by dot products.

After acquiring the attention weights of channels, we multiply them and the original feature maps to update the feature maps. It is denoted as:

\[
\begin{align*}
o_{ac} &= \alpha^c \odot h \\
o_{mc} &= \alpha^m \odot h
\end{align*}
\]

where \(\odot\) is the element-wise multiplication. By doing so, the network will focus on the channels which are more semantically related to the given text description.

Meanwhile, the results of global average pooling and global max pooling are fused through concatenation. Specifically,

\[
o_c = \sigma(W_c[o_{ac}; o_{mc}])
\]

where \(W_c\) is implemented as 1×1 convolution, \(\sigma\) is a nonlinear function, such as ReLU.

To stabilize the learning, we further apply an adaptive residual connection [125] to generate the final result. It is defined as follows:

\[
y_c = \gamma_c \ast o_c + h
\]

where \(\gamma_c\) is a learnable parameter which is initialized as 0.

As can be seen from above, our designed channel-aware attention model is a fine-tuning module based on channel information and text features. Moreover, it is applied on each scale of feature maps to improve the semantic consistency of generated samples at the generative stage.

### 2.4.3 Pixel-aware attention module

An image is composed of correlated pixels which are of central importance for the quality and semantic consistency of synthesized samples. Thus, we propose a pixel-aware attention module to effectively model the relationships between pixels and the given natural-language description and to make the important pixels receive more attentions from the generator.
2.4 PROPOSED METHODOLOGY

Figure 2.5: Overview of the proposed pixel-aware attention module. SAP and SMP denote average pooling and max pooling in the spatial dimension, respectively.

The framework of our pixel-aware attention module is illustrated in Fig. 2.5. Given the feature map \( \hat{h} \) and the global sentence vector \( s \), we first exploit average pooling and max pooling in the spatial dimension to process \( \hat{h} \). Specifically,

\[
e_a = \text{SAP}(\hat{h}) \tag{2.19}
\]

\[
e_m = \text{SMP}(\hat{h}) \tag{2.20}
\]

where SAP and SMP represent average pooling and max pooling in the spatial dimension, respectively. \( e_a \in \mathbb{R}^{1 \times H \times W} \) and \( e_m \in \mathbb{R}^{1 \times H \times W} \) are the new feature maps.

Then, \( s \) is adopted as the key and the value:

\[
k_p = W_{kp}s, v_p = W_{vp}s \tag{2.21}
\]

where \( W_{kp} \) and \( W_{vp} \) are the learnable matrices.

After that, we compute the dot products of the new feature maps and the key to get the semantic similarity \( \tilde{\alpha}_a^p \in \mathbb{R}^{(H \times W) \times D} \) and \( \tilde{\alpha}_m^p \in \mathbb{R}^{(H \times W) \times D} \) between pixels and the global sentence vector. Furthermore, the attention weights are calculated through a softmax function on the dot products of the semantic similarity and the value. It is defined as:

\[
\tilde{\alpha}_a^p = e_a \cdot k_p^T,\tilde{\alpha}_m^p = e_m \cdot k_p^T \tag{2.22}
\]

\[
\alpha_a^p = \text{softmax}(\tilde{\alpha}_a^p \cdot v_p) \tag{2.23}
\]

\[
\alpha_m^p = \text{softmax}(\tilde{\alpha}_m^p \cdot v_p) \tag{2.24}
\]
where $\alpha_p^a \in \mathbb{R}^{H \times W \times 1}$ and $\alpha_m^p \in \mathbb{R}^{H \times W \times 1}$ represent the final attention weights of pixels for average pooling and max pooling, respectively.

Next, same as the channel-aware attention module, we perform a matrix multiplication between the attention weights and the original feature maps to derive the new features $o_{ap}$ and $o_{mp}$:

$$o_{ap} = \alpha_p^a \odot \hat{h}$$
$$o_{mp} = \alpha_p^m \odot \hat{h}$$ (2.25) (2.26)

In addition, we concatenate $o_{ap}$ and $o_{mp}$ and apply a nonlinear function $\sigma$ to compute the result $o_p$. Finally, an adaptive residual connection [125] is utilized to integrate $\hat{h}$ and $o_p$. This process is denoted as:

$$o_p = \sigma(W_p [o_{ap}; o_{mp}])$$ (2.27)
$$y_p = \gamma_p \ast o_p + \hat{h}$$ (2.28)

where $W_p$ is implemented as $1 \times 1$ convolution, $\sigma$ is a nonlinear function such as ReLU and $\gamma_p$ is a learnable parameter which is initialized as 0.

### 2.4.4 Conditional Adaptive Instance-Layer Normalization

In order to stabilize the training of a GAN [30], most existing text-to-image generation models [55], [83], [118], [122], [140] employ Batch Normalization (BN) [91] applying the normalization to a whole batch of generated images instead for single ones. However, the convergence of BN heavily depends on the size of a batch [60]. Furthermore, the advantage of BN is not obvious for text-to-image generation, since each synthesized image is more pertinent to the given text description and the feature map itself. To this end, Conditional Adaptive Instance-Layer Normalization (CAdaILN) is designed to perform the normalization in the layer and channel of the feature map $f$ and modulate the normalized feature map $\hat{f}$ with the linguistic cues captured from the global sentence vector $s$, illustrated in Fig. 2.6. More concretely, we adopt two fully-connected layers $W_1$ and $W_2$ to transform the sentence vector $s$ into the linguistic cues $\gamma \in \mathbb{R}^{1 \times 1 \times C}$ and $\beta \in \mathbb{R}^{1 \times 1 \times C}$. Moreover, we normalize the visual feature map with Instance Normalization (IN) and Layer Normalization (LN). After that, the normalized feature map $\hat{f}$ is acquired via the adaptive sum of the IN output
2.4 PROPOSED METHODOLOGY

2.4.5 Visual loss

To ensure that generated samples and real images have similar color distributions and shape, we propose a new type of visual loss for the generator which is illustrated in Fig. 2.3. The visual loss plays a vital role in improving the quality
and resolution of finally-generated images. It is based on the image features of the real image $I$ and the generated sample $\hat{I}$ and defined as:

$$L_{\text{vis}} = |f(I) - f(\hat{I})|_1 \quad (2.32)$$

where $f(I)$ and $f(\hat{I})$ denote the image features of the real image and the synthesized image. They are achieved by the feature extractor which consists of 5 downsampling residual blocks, 1 residual block and 1 convolutional layer. We impose the L1 loss to minimize the distance between these two image features. To the best of our knowledge, we are the first to present this type of visual loss and apply it in the task of text-to-image generation.

### 2.4.6 Objective function

#### Adversarial loss.

An adversarial loss is employed to match the generated samples to the input textual descriptions. Inspired by [105], [125], we utilize the hinge objective [62] to stabilize the training of a cGAN instead of the vanilla GAN objective. The adversarial loss for the discriminator is formulated as:

$$L^D_{\text{adv}} = \mathbb{E}_{x \sim p_{\text{data}}} [\max(0, 1 - D(x, s))] \\
+ \frac{1}{2} \mathbb{E}_{x \sim p_G} [\max(0, 1 + D(\hat{x}, s))] \\
+ \frac{1}{2} \mathbb{E}_{x \sim p_{\text{data}}} [\max(0, 1 + D(x, \hat{s}))] \quad (2.33)$$

where $s$ is a given text description, $\hat{s}$ is a mismatched natural-language description.

The corresponding generator loss is:

$$L^G_{\text{adv}} = \mathbb{E}_{x \sim p_G} [D(x, s)] \quad (2.34)$$

#### Matching-aware zero-centered gradient penalty (MA-GP) loss.

To enhance the quality and semantic consistency of generated samples, we adopt the MA-GP loss [105] for the discriminator. The MA-GP loss applies gradient penalty to real pictures and given textual descriptions. It is defined as follows:

$$L_{M} = \mathbb{E}_{x \sim p_{\text{data}}} [(\|\nabla_x D(x, s)\|_2 + \|\nabla_s D(x, s)\|_2)^p] \quad (2.35)$$

#### Generator objective.

The generator loss comprises an adversarial loss $L^G_{\text{adv}}$ and a visual loss $L_{\text{vis}}$:

$$L_G = L^G_{\text{adv}} + \lambda_1 L_{\text{vis}} \quad (2.36)$$
Table 2.1: The details of our discriminator network. ‘Conv’ denotes a convolution layer and ‘BN’ denotes batch normalization.

<table>
<thead>
<tr>
<th>Layers</th>
<th>Output shape</th>
<th>Layer Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv</td>
<td>(32, 256, 256)</td>
<td>Conv-(N32, K3, S1, P1)</td>
</tr>
<tr>
<td>Layer I</td>
<td>(64, 128, 128)</td>
<td>Conv-(N64, K4, S2, P1), BN, Leaky-ReLU, Conv-(N64, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
<tr>
<td>Layer II</td>
<td>(128, 64, 64)</td>
<td>Conv-(N128, K4, S2, P1), BN, Leaky-ReLU, Conv-(N128, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
<tr>
<td>Layer III</td>
<td>(256, 32, 32)</td>
<td>Conv-(N256, K4, S2, P1), BN, Leaky-ReLU, Conv-(N256, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
<tr>
<td>Layer IV</td>
<td>(512, 16, 16)</td>
<td>Conv-(N512, K4, S2, P1), BN, Leaky-ReLU, Conv-(N512, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
<tr>
<td>Layer V</td>
<td>(512, 8, 8)</td>
<td>Conv-(N512, K4, S2, P1), BN, Leaky-ReLU, Conv-(N512, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
<tr>
<td>Layer VI</td>
<td>(512, 4, 4)</td>
<td>Conv-(N512, K4, S2, P1), BN, Leaky-ReLU, Conv-(N512, K3, S1, P1), BN, Leaky-ReLU</td>
</tr>
</tbody>
</table>

**Discriminator objective.** The final objective function of the discriminator is defined as follows:

\[ \mathcal{L}_D = \mathcal{L}_D^{adv} + \lambda_2 \mathcal{L}_M \]  \hspace{1cm} (2.37)

2.4.7 Discriminator network

As seen in Table 2.1, the basic architecture of our discriminator network comprises one convolutional layer with strides 1 kernel size 3 padding 1 followed by six adaptively successive residual blocks. Each block consists of two convolutional layers, where the first layer with strides 2 kernel size 4 padding 1 is used to reduce the dimension to half of the input feature map and the second one with strides 1 kernel size 3 padding 1 aims to further distill image features. After each convolutional layer, BN and Leaky ReLU activation with a slope of 0.2 are stacked to aid the training. The number of filters for residual blocks are 64 128 256 512 512 512, respectively. In order to stabilize learning, we also introduce a residual connection into each block. In the process of training, an image is fed into the basic discriminator network to extract the image features (512, 4, 4) which are combined with the sentence vector as the joint embeddings. After that, the joint features are passed through two convolutional layers to gain the final conditional score that is utilized to determine whether the input text-image pair is real or fake.
2.5 EXPERIMENTS

In this section, we carry out a set of experiments on the CUB bird [108] and MSCOCO [63] data sets, in order to quantitatively and qualitatively evaluate the effectiveness of the proposed DTGAN. The previous state-of-the-art cGAN-based text-to-image synthesis models [55], [118], [127], [140] are compared with our approach. Afterwards, we analyze the significant components of our designed framework.

2.5.1 Experimental settings

Datasets. Two popular data sets in text-to-image generation, i.e., CUB bird and MSCOCO data sets, are employed to test our method.

- CUB bird. The CUB bird data set consists of 11,788 images, where 8,855 images belong to the training set and the other 2,933 images belong to the test set. Each picture contains 10 sentences for text descriptions. We preprocess the CUB bird data set using the method in StackGAN [126].
- MSCOCO. The MSCOCO data set is a more challenging data set comprising 123,287 images which are divided into 82,783 training images and 40,504 test images. Each picture has 5 human annotated captions.

Implementation details. For the text encoder, the dimension of the sentence vector is set to 256 and the length of words is set to 18. We implement our model using PyTorch [81]. In the experiments, the network is trained using Adam optimizer [49] with $\beta_1 = 0.0$ and $\beta_2 = 0.9$. We follow the two timescale update rule (TTUR) [38] and set the learning rate of the generator and the discriminator to 0.0001 and 0.0004, respectively. The dimension of the noise vector is set to 100. The batch size is set to 24 on the CUB bird data set and to 16 on the MSCOCO data set. The hyperparameters $p$, $\lambda_1$ and $\lambda_2$ are set to 6, 0.1 and 2, respectively. All the experiments are performed on a single NVIDIA Tesla V100 GPU (32 GB memory).

Evaluation metrics. Inception score (IS) [89] and Fréchet inception distance (FID) [103] score are extensively employed in the assessment of text-to-image generation. We adopt these two indexes as the quantitative evaluation measure. Meanwhile, we perform a human test to evaluate the quality and semantic consistency of generated images.

- IS. The IS is acquired via the KL divergence between the conditional class distribution and the marginal class distribution. It is defined as:

$$ I = \exp(\mathbb{E}_x[D_{KL}(p(y|x) \parallel p(y))]) $$

(2.38)

where $x$ is a generated sample and $y$ is the corresponding label obtained by a pretrained Inception v3 network [103]. The produced samples are split into
multiple groups and the IS is calculated on each group of images, then the average and standard deviation of the score are reported. Higher IS demonstrates better quality among the synthesized images.

For the MSCOCO data set, DF-GAN [105] and ObjGAN [58] argue that the IS fails to evaluate the synthetic samples and can be saturated, even over-fitted. Consequently, we do not compare the IS on the MSCOCO data set.

- **FID.** The FID computes the Fréchet distance between the distribution of generated pictures and the distribution of true data. A lower FID means that the synthetic samples are closer to the corresponding real images. We use a pretrained Inception v3 network to achieve the FID score.

It should be noted that we produce 30,000 pictures from unseen textual descriptions for the IS and FID.

### 2.5.2 Comparison with state of the art

**Quantitative results.** We compare our model with prior state-of-the-art cGAN-based approaches in text-to-image synthesis on the CUB bird and MSCOCO data sets. Table 2.2 reports the IS of our proposed DTGAN and other compared methods on the CUB bird data set. We can observe that our model has the best score, significantly improving the IS from 4.75 to 4.88 on the CUB bird data set. Experimental results demonstrate that DTGAN can generate visually realistic images with higher quality than previous state-of-the-art models.

The comparison between our method, StackGAN++ [127] and AttnGAN [118] with respect to FID on the CUB bird and MSCOCO data sets is shown in Table 2.3.
Table 2.3: The FID of StackGAN++ [127], AttnGAN [118] and our model on the CUB bird and MSCOCO data sets. The best results are in bold.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>StackGAN++</th>
<th>AttnGAN</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUB</td>
<td>26.07</td>
<td>23.98</td>
<td><strong>16.35</strong></td>
</tr>
<tr>
<td>COCO</td>
<td>51.62</td>
<td>35.49</td>
<td><strong>23.61</strong></td>
</tr>
</tbody>
</table>

We can see that DTGAN achieves a remarkably lower FID than compared approaches on both data sets, which indicates that the distribution of generated data is closer to the distribution of real data. Specifically, we impressively reduce the FID from 35.49 to 23.61 on the challenging MSCOCO data set and from 23.98 to 16.35 on the CUB bird data set.

**Qualitative results.** In addition to quantitative experiments, we perform qualitative comparison with StackGAN++ [127] and AttnGAN [118] on both data sets. The results are illustrated in Fig. 2.7, which indicates that DTGAN is able to produce high-quality and semantically consistent pictures according to natural-language descriptions. As can be observed in Fig. 2.7 (a), the samples synthesized by StackGAN++ and AttnGAN lack text-relevant objects (1st, 2nd, 3rd, 5th, 7th and 8th column), the backgrounds are unclear and inconsistent with the given text descriptions (1st, 3rd, 5th, 6th, 7th and 8th column) and the colors are rough (4th, 7th and 8th column) on the challenging MSCOCO data set. Furthermore, the details of birds generated by these two approaches are lost (2nd, 3rd, 4th, 5th, 6th and 8th column), the shape is strange (1st, 3rd, 4th and 7th column) and the colors are even wrong (3rd and 6th column) on the CUB bird data set, which is shown in Fig. 2.7 (b). However, DTGAN generates more clear and visually plausible images than StackGAN++ and AttnGAN, verifying the superiority of DTGAN. For instance, as shown in the 1st column of Fig. 2.7 (b), owing to the successful application of the visual loss, a long-wingspan bird with vivid shape is produced by DTGAN, whereas it is too hard for StackGAN++ and AttnGAN to generate this kind of bird. In the meantime, the birds generated by DTGAN have more details and richer colors compared to StackGAN++ and AttnGAN in the 2nd, 3rd, 4th, 6th, 7th and 8th column, since DTGAN armed with channel-aware and pixel-aware attention modules is able to generate high-resolution images which semantically align with given text descriptions. More importantly, our method also yields high-quality and visually realistic results on the challenging MSCOCO data set. For example, the number of the skiers and surfers is correct, the backgrounds are reasonable and people in the images are clear in the 1st, 2nd, 5th, 6th, 7th and 8th column of Fig. 2.7 (a). Moreover, the beach, the sea and the pizza in the 3rd, 4th and 8th column look realistic. Generally, these qualitative results confirm the effectiveness of the presented DTGAN.
Furthermore, to validate the sensitivity and diversity of DTGAN, we yield birds by modifying just one word in the given text description or changing the random latent vector. As can be seen in Fig. 2.8, the generated birds are similar but have different poses and shape for the same description. When we change the color attribute in the natural-language description, the proposed DTGAN further produces semantically consistent birds according to the modified text. It means that our approach has the ability to capture the modified part of the text description but produces slightly diverse birds based on a single text description and different injected latent vectors. The lack-of-diversity issue will be addressed in the next chapter.

**Human evaluation.** In order to evaluate the image quality and semantic consistency of StackGAN++ [127], AttnGAN [118] and DTGAN, we also conduct
Figure 2.8: Generated images of DTGAN by changing the color attribute value in the input text description, for four random draws.

Table 2.4: Human test results (ratio of 1st) of StackGAN++ [127], AttnGAN [118] and our model on the CUB bird and MSCOCO data sets. The best results are in bold.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>StackGAN++</th>
<th>AttnGAN</th>
<th>Ours</th>
</tr>
</thead>
<tbody>
<tr>
<td>CUB</td>
<td>8.1%</td>
<td>15.3%</td>
<td>76.6%</td>
</tr>
<tr>
<td>COCO</td>
<td>6.2%</td>
<td>11.2%</td>
<td>82.6%</td>
</tr>
</tbody>
</table>

a human test on the CUB bird and MSCOCO data sets. We randomly select 100 images from both data sets, respectively. Users are asked to choose the best image from three given images according to the image quality and text description. Furthermore, the final results are calculated by two judges for fairness. As shown in Table 2.4, our method outperforms AttnGAN by 61.3% on the CUB bird data set and 71.4% on the MSCOCO data set, validating the superiority of DTGAN.

2.5.3 Component analysis

In this section, we perform an extensive ablation study on the CUB bird data set, so as to evaluate the contributions from different components of DTGAN. The novel components in our model include a channel-aware attention module (CAM), a pixel-aware attention module (PAM), CAdaILN and a new type of visual loss (VL). We first quantitatively explore the effectiveness of each component by
Table 2.5: Ablation study of DTGAN. CAM, PAM and VL represent the channel-aware attention module, the pixel-aware attention module and the visual loss, respectively. The best results are in bold.

<table>
<thead>
<tr>
<th>ID</th>
<th>Components</th>
<th>IS↑</th>
<th>FID↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>✓ ✓ ✓ ✓ ✓</td>
<td>4.88 ± 0.03</td>
<td>16.35</td>
</tr>
<tr>
<td>2</td>
<td>✓ ✓ ✓ - -</td>
<td>4.72 ± 0.04</td>
<td>19.23</td>
</tr>
<tr>
<td>3</td>
<td>✓ ✓ - - ✓</td>
<td>4.11 ± 0.04</td>
<td>25.24</td>
</tr>
<tr>
<td>4</td>
<td>✓ - ✓ ✓ ✓</td>
<td>4.71 ± 0.05</td>
<td>21.69</td>
</tr>
<tr>
<td>5</td>
<td>- - ✓ ✓ ✓</td>
<td>4.60 ± 0.07</td>
<td>22.95</td>
</tr>
<tr>
<td>6</td>
<td>- - - ✓ ✓</td>
<td>4.54 ± 0.04</td>
<td>23.72</td>
</tr>
</tbody>
</table>

removing the corresponding part in the proposed DTGAN step by step, i.e., 1) DTGAN, 2) DTGAN without the VL, 3) DTGAN without CAdaILN, 4) DTGAN without the PAM, 5) DTGAN without the CAM, 6) DTGAN without both the CAM and PAM. All the results are reported in Table 2.5.

By comparing Model 1 (DTGAN) with Model 2 (removing the VL), the VL significantly improves the IS from 4.72 to 4.88 and reduces the FID by 2.88 on the CUB bird data set, which demonstrates the importance of adopting the VL in DTGAN. By exploiting CAdaILN in DTGAN, Model 1 performs better than Model 3 (removing CAdaILN) on the IS and FID by 0.77 and 8.89, confirming the effectiveness of the proposed CAdaILN. Both Model 4 (removing the CAM) and Model 5 (removing the PAM) outperform Model 6 (removing both the CAM and PAM), indicating that these two new types of attention modules can help the generator produce more realistic images. Furthermore, Model 1 achieves better results than both Model 4 and Model 5, which shows the advantage of combining the CAM and PAM.

Attention modules. To better understand what has been learned by the CAM and PAM during training, we visualize the channel-aware and pixel-aware attention maps for different images in Fig. 2.9. We can see that in the 3rd row, the eyes, beaks, legs and wings of birds are highlighted by the channel-aware attention maps. Meanwhile, in the 4th row, the pixel-aware attention maps highlight most important areas of images that contain the branches and the whole bodies of birds. This suggests that the CAM helps the generator focus on the crucial parts of birds, while the PAM guides the generator to refine the overall visual content. In other words, the discriminative regions of generated samples are fine-tuned and facilitated by our attention modules. Note that some texture background regions that are related to the conditional contexts might be focused.

Visual loss. To balance the trade-off between image quality and semantic consistency, we investigate the hyper-parameter $\lambda_1$ by changing its value in the
Generated Images

Figure 2.9: Visualization of the channel-aware (detailed features) and pixel-aware (global shape) attention maps.

Table 2.6: Evaluation of DTGAN for different values of $\lambda_1$ which is the weight of the visual loss (VL) in the generator. The best result is in bold.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Values</th>
<th>IS ↑</th>
<th>FID ↓</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda_1$</td>
<td>0.05</td>
<td>4.74 ± 0.05</td>
<td>18.15</td>
</tr>
<tr>
<td></td>
<td>0.10</td>
<td><strong>4.88 ± 0.03</strong></td>
<td><strong>16.35</strong></td>
</tr>
<tr>
<td></td>
<td>0.15</td>
<td>4.82 ± 0.06</td>
<td>16.75</td>
</tr>
<tr>
<td></td>
<td>0.20</td>
<td>4.59 ± 0.04</td>
<td>20.91</td>
</tr>
<tr>
<td></td>
<td>0.30</td>
<td>4.70 ± 0.06</td>
<td>20.28</td>
</tr>
</tbody>
</table>

objective function. We test the value of $\lambda_1$ among 0.05, 0.10, 0.15, 0.20 and 0.30. The results are listed in Table 2.6. We can observe that the best performance is achieved on the CUB bird data set when $\lambda_1$ is set to 0.1. We therefore set $\lambda_1$ to 0.1 in the experiments.

In addition, we conduct an ablation study to validate the effectiveness of the visual loss (VL). The visual comparison between DTGAN and our model without the VL is shown in Fig. 2.10. We can see that, in the first four columns, DTGAN without the VL fails to generate long-wingspan birds with reasonable shape and vivid wings. In the meantime, the proposed model without the VL synthesizes the blue birds which have rough colors and lack colorful details in the last four columns. However, DTGAN produces realistic long-wingspan birds that have semantically consistent shape and colors, while also yielding blue birds with vivid details and rich colors. This indicates that the VL has the ability to potentially
ensure the quality of generated samples, e.g., the shape and color of the objects in an image.

CAdaILN. To further verify the benefits of CAdaILN, we conduct an ablation study on normalization functions. We first design a baseline model by removing CAdaILN from DTGAN. Then, we compare the variants of normalization layers. Note that BN conditioned on the global sentence vector (BN-sent) and BN conditioned on the word vectors (BN-word) are based on semantic-conditioned BN in SDGAN [122] and the CAdaILN function with the word vectors (CAdaILN-word) is achieved through the word-level normalization method in SDGAN. The results of the ablation study are shown in Table 2.7. It can be observed that by comparing Model 2 with Model 4 and Model 3 with Model 5, CAdaILN significantly outperforms the BN layer whether using the sentence-level cues or the word-level cues. Moreover, by comparing Model 4 with Model 5, CAdaILN with the global sentence vector performs better than CAdaILN-word by improving the IS from
4.71 to 4.88 and reducing the FID from 19.08 to 16.35 on the CUB bird data set, since sentence-level features are easier to train in our generator network than word-level features. The above analysis demonstrates the effectiveness of our designed CAdaILN.

2.6 CONCLUSION

In this chapter, we propose the Dual-Attention Generative-Adversarial Network (DTGAN), a novel framework for text-to-image generation, to generate perceptually realistic images which semantically align with given text descriptions, only employing a single generator/discriminator pair. More specifically, DTGAN exploits two new types of sentence-level attention modules: a channel-aware attention module and a pixel-aware attention module, to guide the generator to focus more on the text-relevant channels and pixels. In addition, Conditional Adaptive Instance-Layer Normalization (CAdaILN) is designed to adopt the linguistic cues to flexibly control the amount of change in shape and texture, strengthening visual-semantic representation and complementing modulation modules. To further enhance the quality of generated images, we design a new type of visual loss which computes the $L_1$ loss between the features of generated samples and real images. DTGAN achieves promising results on the CUB bird and MSCOCO data sets, which confirms the superiority of our proposed method. However, the improved visual quality comes with an apparent reduction in the variation of generated images. Therefore, Chapter 3 will be directed at mitigating this phenomenon by presenting an architecture that is better able to retain diversity in the generated images.