Text to Image Synthesis using DF-GAN

Tejas Jadhav Nakul Gupta Shraddha Gadale

Abstract

Generating realistic images from text descriptions is a challenging task. Many existing GAN-based methods rely on stacked architectures, which often lead to issues such as entangled generators, limited semantic consistency, and high computational costs due to cross-modal fusion. DF-GAN addresses these challenges with a simpler and more effective approach, including a one-stage backbone for direct high-resolution image generation, a target-aware discriminator to enhance text-image alignment, and a deep fusion block for better feature integration. This method is more efficient and achieves superior performance compared to existing approaches.

1. Introduction

The most important application of the Generative Adversarial Network is text to image generation. GAN aims to generate the image from the given text description. The challenging part of text to image generation is authenticity of the generated image and consistency between text description and generated image.

Most of the models using GANs uses stacked architecture to make GAN stable. They mosly use cross model attention to fuse text and image features , DAMSM network , cycle consistency , or siamese network .

Existing text-to-image models face three challenges. First, stacked architectures with multiple generators which result in entanglements, producing images that appear as inconsistent combinations of features. Second, the auxiliary networks used to ensure semantic consistency, such as DAMSM and Siamese networks, are fixed during adversarial training, allowing the generator to fool them with adversarial features, which weakens their supervision. Lastly, cross-modal attention, used to fuse text and image features, is computationally expensive and limited to low resolutions (e.g., 64×64 and 128×128), reducing the effectiveness of the fusion process and making it difficult to scale to higher-resolution image synthesis.

To solve the challenges in existing text-to-image models, Instead of using stacked generators, DF-GAN uses a single generator with hinge loss and residual networks. This approach makes training more stable and creates high-resolution images directly. MA-GP helps the model focus on generating images that match the text, while the One-Way Output speeds up training.

DFBlocks use lightweight operations to blend text information into image features more effectively. By stacking DFBlocks across different image scales, DF-GAN ensures a deeper and more thorough fusion of text and visuals. These improvements allow DF-GAN to generate clear, high-quality images that align with the given text, performing better than existing models on difficult datasets.

1.1. Related Work

Generative Adversarial Networks are widely used for modeling real-world data by training a generator and discriminator in a min-max optimization process. StackGAN created high-resolution images using multiple generators and discriminators, combining text information with random noise. AttnGAN added cross-modal attention to generate more detailed images. DM-GAN refined blurry images with a memory network. More recently, transformer-based methods tokenize images and text for training, achieving excellent results on complex image synthesis tasks.

DF-GAN is different from these methods, It uses a one-stage backbone to generate high-resolution images directly. It also includes a Target-Aware Discriminator to improve text-image alignment without extra networks. Finally, DF-GAN uses Deep Fusion Blocks (DFBlocks) to deeply and effectively combine text and image features. Compared to previous methods, DF-GAN is simpler, faster, and better at generating realistic, text-matching images.

2. Methods

2.1. Overview of DF-GAN

DF-GAN has three important parts generator, discriminator, and pre-trained text encoder, as shown in Figure 2. The generator takes a sentence vector encoded by the text encoder and a noise vector sampled from a Gaussian distribution to ensure image diversity. The noise vector is first processed by a fully connected layer and then reshaped. Next, a series of UPBlocks, consisting of an upsampling layer, a resid-

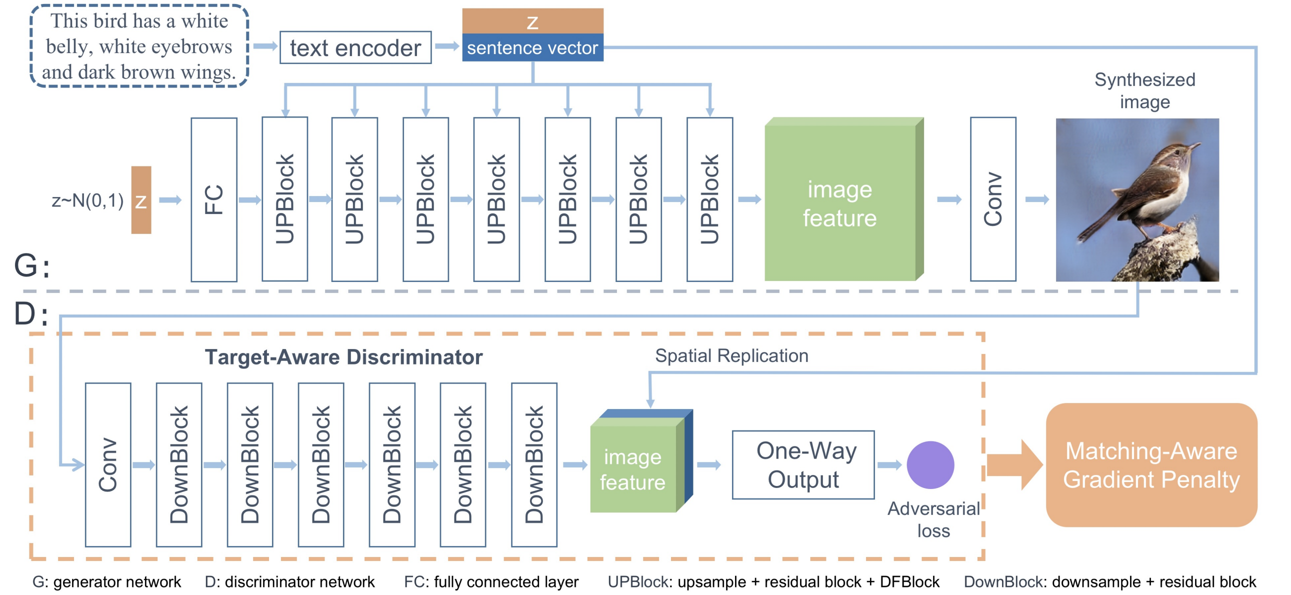

Figure 1. DF-GAN uses a single generator-discriminator pair to produce high-resolution, text-aligned images. Text and visual features are fused via DFBlocks in UPBlocks, while MA-GP and One-Way Output enhance realism and semantic consistency.

ual block , and DFBlocks, are used to generate and refine image features while integrating text information. Finally, a convolutional layer transforms these features into a final image.

The discriminator processes images through a series of DownBlocks, converting them into feature representations. The sentence vector is then replicated and concatenated with these features. The discriminator evaluates both the visual quality and the semantic consistency of the inputs by predicting an adversarial loss. This encourages the generator to create images that are both realistic and text-matching.

The text encoder is a bidirectional long-short-term memory (LSTM) network that extracts semantic features from the input text. This pre-trained encoder is directly adopted from AttnGAN , ensuring high-quality text feature extraction.

2.2. One stage Method

Previous text-to-image GANs often use stacked architectures to generate high-resolution images by refining low-resolution outputs. However, stacked architectures introduce entanglements between generators, leading to final images that appear as a combination of fuzzy shapes and mismatched details Figure 1(a).

To address this, a one-stage text-to-image backbone was introduced. This produces high-resolution images directly using a single generator and discriminator; we do not use stacked generators to avoid entanglements. The adversarial training process is stabilized using hinge loss . To produce high-resolution images directly from noise vectors, we have to use more layers than stacked architectures. To effectively train deeper networks, we use residual networks .

- Discriminator loss (L_D): Encourages the discriminator to distinguish between real, generated, and mismatched data.

$$\begin{aligned} L_D = & -E_{x\sim P_r}[\min(0, -1 + D(x, e))] \\ & -\frac{1}{2}E_{G(z)\sim P_g}[\min(0, -1 - D(G(z), e))] \\ & -\frac{1}{2}E_{x\sim P_{\text{mis}}}[\min(0, -1 - D(x, e))] \end{aligned}$$

- Generator Loss (L_G): Guides the generator to create images that the discriminator perceives as real and text-aligned.

$$L_G = -E_{G(z)\sim P_g}[D(G(z), e)]$$

Here, z represents a noise vector sampled from a Gaussian distribution, e is the sentence vector, and P_r, P_g, and P_{mis} denote the real, generated and mismatched data distributions, respectively.

3. Discriminator

3.1. MA-GP

The Matching-Aware Gradient Penalty is a strategy designed to improve text-image semantic consistency. It builds upon the concept of gradient penalty in unconditional image generation, where the discriminator's loss landscape is smoothed around real data points. This smoothing helps the generator converge toward creating realistic images.

- In text-to-image generation, the discriminator evaluates four types of inputs: synthetic images with matching text (fake, match)
- synthetic images with mismatched text (fake, mismatch)
- real images with matching text (real, match)
- real images with mismatched text (real, mismatch)

MA-GP applies the gradient penalty specifically to real images with matching text (real, match). This helps the discriminator create a smoother loss surface for text-matching real data, guiding the generator to produce images that better align with the input text.

The mathematical formulation for the discriminator loss (L_D) and generator loss (L_G) with MA-GP is:

Discriminator Loss (L_D):

$$L_D = -E_{x\sim P_r}[\min(0, -1 + D(x, e))]$$

$$-\frac{1}{2}E_{G(z)\sim P_g}[\min(0, -1 - D(G(z), e))]$$

$$-\frac{1}{2}E_{x\sim P_{\text{mis}}}[\min(0, -1 - D(x, e))]$$

$$+kE_{x\sim P_r}[(\|\nabla_x D(x, e)\| + \|\nabla_e D(x, e)\|)^p]$$

Generator Loss (L_G):

$$L_G = -E_{G(z)\sim P_g}[D(G(z), e)]$$

Here, k and p are hyperparameters controlling the strength of the gradient penalty. x represents real image data, z is the noise vector, and e is the sentence vector. P_r, P_g, and P_{mis} denote the real, generated, and mismatched data distributions, respectively.

By applying MA-GP as a regularization technique, the model better converges to the real, text-matching data and generates more semantically aligned images. Additionally, since the discriminator is jointly trained with the generator, it prevents the generator from synthesizing adversarial features that could weaken semantic consistency. Moreover, MA-GP does not rely on extra networks, and its only additional computation is gradient summation, making it more efficient compared to methods that use fixed auxiliary networks.

3.2. One Way Output

In previous GANs , the discriminator uses a two-way output (Figure 4 (a)), where we determine whether the image is real or fake (conditional loss) and evaluate the semantic consistency of the text image by combining the features of the image with the vector of sentences (conditional loss). These two losses are then computed separately, Which is ineffective and slows down generator convergence. This happens because the gradients of unconditional loss (β) and conditional loss (α) are simply summed, leading to a final gradient direction that deviates from the target (real and text-matching data). This misalignment makes it harder for the generator to achieve semantic consistency and slows its training.

To address this, One-Way Output computes two separate losses, discriminator concatenates the image feature and sentence vector, and outputs a single adversarial loss through two convolution layers. Because of this the gradient (α) directly points to the target data (real and matching images), optimizes, and accelerates generator convergence.

4. Fusion Block

The Text-Image Fusion Block (DFBlock) enhances the text-image fusion process. DFBlock ensures deeper and comprehensive fusion, enabling the generator to fully utilize the text information during image generation.

The generator consists of 7 UPBlocks, each containing two DFBlocks. Each DFBlock stacks multiple Affine Transformations and ReLU layers to fuse text and image features. The Affine transformation uses two multilayer perceptrons (MLP) to predict channelwise scaling (γ) and shifting parameters (θ) from the sentence vector (e).

$$\gamma = \text{MLP}_1(e), \quad \theta = \text{MLP}_2(e).$$

Given an input feature map $X \in R^{B\times C\times H\times W}$, the channelwise scaling and shifting are applied as:

$$\text{AFF}(x_i|e) = \gamma_i \cdot x_i + \theta_i,$$

where AFF denotes the Affine Transformation, x_i is the i-th channel of the feature map, and γ_i, θ_i are the scaling and shifting parameters.

To improve the fusion process, ReLU layer between two Affine Transformations was added, introducing nonlinearity and expanding the conditional representation space. This allows the generator to better map different images to unique representations based on text descriptions.

DFBlock builds on ideas from Conditional Batch Normalization (CBN) and Adaptive Instance Normalization (AdaIN)

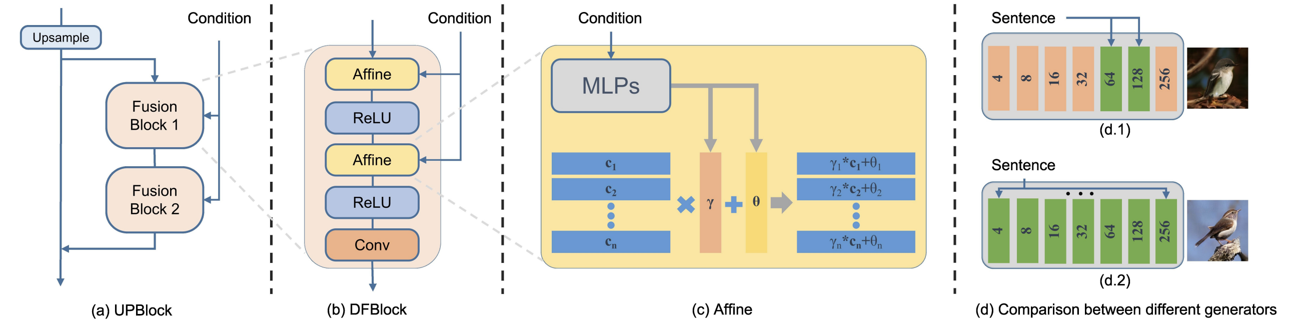

Figure 2. (a) A typical UPBlock upsamples image features and integrates text through two Fusion Blocks. (b) DFBlock includes two Affine layers, ReLU activations, and a Convolution layer. (c) Affine Transformation scales and shifts features based on text inputs. (d) Comparison: (d.1) cross-modal attention vs. (d.2) DFBlock for efficient text-image fusion.

, which also use Affine Transformations. However, unlike CBN and AdaIN, which rely on normalization layers that reduce feature diversity, DFBlock removes normalization. This ensures better differentiation between samples, making it more effective for conditional image generation.

By deepening the fusion process, DFBlock provides two key benefits:

- This allows the generator to use the text information more effectively during fusion.
- It enlarges the representation space, helping the generator produce diverse and semantically consistent images for various text descriptions.

5. Experiments

We evaluated DF-GAN on the CUB bird dataset . Contains 11,788 bird images from 200 species, each with ten text descriptions.

The network is trained using Adam Optimizer with parameters $\beta_1 = 0.0$ and $\beta_2 = 0.9$. The learning rate is set according to the two-timescale update rule (TTUR) set as Generator at 0.0001 and Discriminator at 0.0004.

We use the Inception Score (IS) and the Frechet Inception Distance (FID) for performance evaluation in which the Inception Score (IS) measures the KL divergence between the conditional and marginal distributions. A higher IS indicates better image quality and clear class representation, and Frechet Inception Distance (FID) computes the Frechet distance between the distributions of generated and real-world images in the feature space of a pre-trained Inception v3 network. Lower FID values indicate more realistic images.

For both metrics, we generated 10,000 images at 256×256 resolution using randomly selected text descriptions from the test set.

Model	**CUB IS ↑**	**FID ↓**	**NoP ↓**
StackGAN	3.70	-	-
StackGAN++	3.84	-	-
AttnGAN	4.36	23.98	230M
DM-GAN	4.75	16.09	46M
DAE-GAN	4.42	15.19	98M
TIME	4.91	14.30	120M
DF-GAN	~ 5.00	~ 14.90	19M

Table 1. Comparison of models on the CUB dataset. IS indicates Inception Score (higher is better), FID indicates Fréchet Inception Distance (lower is better), and NoP indicates the number of parameters.

6. Evaluation

We compare DF-GAN with state-of-the-art methods, including StackGAN , AttnGAN , MirrorGAN , SD-GAN , and DM-GAN , as well as newer models like CPGAN , XMC-GAN , and DAE-GAN , which rely on extra knowledge (e.g., pre-trained YOLO-V3 , VGG-19 , and BERT).

DF-GAN achieves competitive results with fewer Number of Parameters (NoP). On the CUB dataset, DF-GAN improves Inception Score (IS) from 4.36 to 5.10 and reduces Frechet Inception Distance (FID) from 23.98 to 14.90, compared to AttnGAN . Against DM-GAN , it increases IS from 4.75 to 5.0 and decreases FID from 16.09 to 14.90. These results show that DF-GAN is simpler yet more effective, outperforming methods that use additional supervision.

7. Conclusion

This paper introduced DF-GAN, approach for text-to-image generation. The model features a one-stage backbone that generates high-resolution images directly, avoiding the complexities of stacked architectures. We also proposed a target-aware discriminator, combining match-aware gradient penalty (MA-GP) and one-way output, to enhance text-image semantic consistency without relying on extra networks. Additionally, the Deep Text-Image Fusion Block (DFBlock) effectively integrates text and image features for better synthesis.

8. References

1. Goodfellow, I., et al. (2014). ”Generative Adversarial Nets.” Advances in Neural Information Processing Systems.

2. Reed, S., et al. (2016). ”Generative Adversarial Text to Image Synthesis.” Proceedings of the International Conference on Machine Learning.

3. Zhang, H., et al. (2017). ”StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks.” Proceedings of the IEEE International Conference on Computer Vision.

4. Xu, T., et al. (2018). ”AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

5. Zhang, H., et al. (2018). ”StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks.” IEEE Transactions on Pattern Analysis and Machine Intelligence.

6. He, K., et al. (2016). ”Deep Residual Learning for Image Recognition.” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

7. Ioffe, S., & Szegedy, C. (2015). ”Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift.” International Conference on Machine Learning.

8. Huang, X., & Belongie, S. (2017). ”Arbitrary Style Transfer in Real-Time with Adaptive Instance Normalization.” Proceedings of the IEEE International Conference on Computer Vision.

9. Kingma, D. P., & Ba, J. (2015). ”Adam: A Method for Stochastic Optimization.” International Conference on Learning Representations.

10. Heusel, M., et al. (2017). ”GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium.” Advances in Neural Information Processing Systems.

11. Lin, T.-Y., et al. (2014). ”Microsoft COCO: Common Objects in Context.” European Conference on Computer Vision.

12. Devlin, J., et al. (2018). ”BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding.” arXiv preprint arXiv:1810.04805.

13. Radford, A., et al. (2021). ”Learning Transferable Visual Models from Natural Language Supervision.” Proceedings of the International Conference on Machine Learning.

14. Ramesh, A., et al. (2021). ”Zero-Shot Text-to-Image Generation.” arXiv preprint arXiv:2102.12092.

15. Zhang, H., et al. (2019). ”Self-Attention Generative Adversarial Networks.” International Conference on Machine Learning.