Article

CapGAN: Text-to-Image Synthesis Using Capsule GANs

Maryam Omar [1], Hafeez Ur Rehman [1,2,*], Omar Bin Samin [3], Moutaz Alazab [2,4], Gianfranco Politano [5] and Alfredo Benso [5]

1 Department of Computer Science, National University of Computing and Emerging Sciences, Hayatabad, Peshawar 24720, Pakistan; p220004@pwr.nu.edu.pk
2 School of Computing and Data Sciences, Oryx Universal College with Liverpool John Moores University, Doha 34110, Qatar; moutaz.a@oryx.edu.qa or m.alazab@bau.edu.jo
3 Center for Excellence in Information Technology, Institute of Management Sciences, Hayatabad, Peshawar 24720, Pakistan; omar.samin@imsciences.edu.pk
4 Department of Intelligent Systems, Faculty of Artificial Intelligence, Al-Balqa Applied University, Al-Salt 19117, Jordan
5 Department of Control and Computer Engineering (DAUIN), Politecnico di Torino, 10129 Turin, Italy; gianfranco.politano@polito.it (G.P.); alfredo.benso@polito.it (A.B.)
* Correspondence: hafeez.r@oryx.edu.qa

Abstract: Text-to-image synthesis is one of the most critical and challenging problems of generative modeling. It is of substantial importance in the area of automatic learning, especially for image creation, modification, analysis and optimization. A number of works have been proposed in the past to achieve this goal; however, current methods still lack scene understanding, especially when it comes to synthesizing coherent structures in complex scenes. In this work, we propose a model called CapGAN, to synthesize images from a given single text statement to resolve the problem of global coherent structures in complex scenes. For this purpose, skip-thought vectors are used to encode the given text into vector representation. This encoded vector is used as an input for image synthesis using an adversarial process, in which two models are trained simultaneously, namely: generator (G) and discriminator (D). The model G generates fake images, while the model D tries to predict what the sample is from training data rather than generated by G. The conceptual novelty of this work lies in the integrating capsules at the discriminator level to make the model understand the orientational and relative spatial relationship between different entities of an object in an image. The inception score (IS) along with the Fréchet inception distance (FID) are used as quantitative evaluation metrics for CapGAN. IS recorded for images generated using CapGAN is 4.05 ± 0.050, which is around 34% higher than images synthesized using traditional GANs, whereas the FID score calculated for synthesized images using CapGAN is 44.38, which is ab almost 9% improvement from the previous state-of-the-art models. The experimental results clearly demonstrate the effectiveness of the proposed CapGAN model, which is exceptionally proficient in generating images with complex scenes.

Keywords: image generation; generative adversarial network (GAN); capsule network; deep learning; convolutional neural network (CNN); skip-thought vectors; inception score (IS); Fréchet inception distance (FID)

1. Introduction

Text-to-image synthesis is the translation of a single sentence directly into pixels [1]. Automatically generating images from a single sentence is a primary problem in computer-aided design (CAD), automatic art generation (AAG) and various other applications. The difficulty of synthesizing images or illustrating visual information from text has gained interest in the research community, but it is far from being solved, in particular for complex scenes. In complex scenes, objects are composed of multiple entities in which the color(s)

and basic shape of each entity can be fully determined separately. Image synthesis from text can facilitate the automatic learning process in the following ways:

1. Once the images are synthesized, any modification in a scene or an image can be implemented by means of text as an input instead of using advanced photo editing tools.
2. Text-to-image synthesis can improve the predictions of object classification problems, as the synthesis model is generating images from scratch, thus, it has good judgment about object features.
3. It will smooth the automatic learning process and art generation of, for example, animated images, clips, movies, etc.
4. The images synthesized using text can also be helpful to generate labeled data for further research.

The problem of generating images from text description is highly multimodal. This means that there can be multiple correct answers for a single input sentence. In terms of image synthesis from text, multimodality suggests that there can be several reasonable and possible configurations of pixels that can correctly, as well as acceptably, exemplify the same text description. For instance, a sentence, "This flower has petals that are white and has a green tip" or "A bird with yellow beak sitting on a tree" can have multiple possible solutions. An illustration of such cases is shown in Table 1, which presents multiple images generated for a given input text.

Table 1. Examples of multiple images generated from a single text statement.

Text	Generated Images
This flower has long red petals with black center.	
A water flower with light yellow petals and yellow pistils in the center.	
This flower has purple petals and a long stigma.	
This flower has rounded white petals which form a bright yellow shape in the center.	
This bird is dark grey in color and has a long wings and a black downward curved beak.	
The bird is a royal blue with black accents on the wings, tail and beak.	
White smiling dog.	

The past research on solving multimodal problems in text-to-image synthesis has focused on various machine-learning-based algorithms, particularly generative adversarial networks (GANs) [2]. Many researchers have made attempts to synthesize images using text for single objects [1,3–5] by plugging conventional convolutional and deconvolutional layers in GANs. However, none of them succeeded in generating coherent images, and the proposed models failed to take into account the spatial and orientational association among diverse entities of an object in an image.

The models based on convolutional layers, e.g., convolutional neural networks (CNN), have provided massive success for several deep learning applications; nonetheless, they have some limitations and drawbacks. For instance, large amounts of data are required for training CNN. In addition, the internal data representation of CNN does not take into account important spatial associations among objects. In order to clarify this, we present an example of a flower object in Figures 1 and 2. For CNN, both Figures 1 and 2 are flowers as CNN does not take into account the spatial and orientational association among different entities of an object. The spatial and orientational association dictates that for an object to

be a flower, the petals should be aligned around the center, the leaves should be connected to the stem and all entities (petals, stem, leaves and stamen) should be connected.

　　　For CNN, both Figures 1 and 2 are flowers, as the mere presence of the entities (petals, stem, leaves and stamen) indicates object existence. However, for a capsule network, Figure 1 is a flower, whereas Figure 2 is not considered to be a flower.
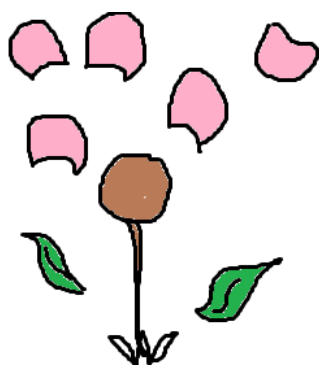

Figure 1. Complete flower.

Figure 2. Disoriented flower.

　　　On the other hand, capsule networks [6] are based on the concept of inverse rendering. In computer graphics, objects are constructed though rendering, which requires some geometric information that specifies where to draw an object, its scale, its angle, along with other spatial information. Capsules in capsule networks are designed to represent vectors or multi-dimensional information, whereas neurons in convolutional neural networks (CNNs) typically operate on scalar values or single-channel data. Therefore, unlike neurons in CNN, the capsules extract the geometric information of an object in an image in the form of vectors and use it for inverse rendering. Therefore, in contrast to CNN, a capsule network easily identifies Figure 1 as a flower, and Figure 2 to be a non-flower object; as in Figure 2, the spatial association between different entities of flowers is not satisfied.

　　　We utilize the capsule network for image synthesis from a given text statement to overcome the problem of global coherent structures in complex scenes. In this work, we suggest an innovative model, named CapGAN, to synthesize images from a given single text statement using GANs. Our model takes as input a single text statement and utilizes skip-thought vectors as text encoders that can produce highly generic fixed length sentence representations [7]. As the synthesis of images from extracted features is highly multimodal, this makes GAN an ideal candidate for image synthesis problems. We feed these fixed length representations to GANs for image synthesis using an adversarial process, in which two models are trained at the same time, namely: generator (G) and discriminator (D). The model G generates fake images, while the model D tries to predict what the sample is from training data rather than generated by G. The conceptual novelty of this work

lies in integrating capsules at the discriminator level to make the model understand the orientational and relative spatial relationship among various diverse entities of an object in an image. CapGAN is trained and evaluated on the Oxford-102 dataset [8] for flowers, Caltech-UCSD 200 [9] for birds and ImageNet [10] for images of dogs. Our model is evaluated using the most widely used inception score (IS) and Fréchet inception distance (FID) measures for the image synthesis problems. The bench-marked results affirm the usefulness of capsule networks to capture and regenerate orientation, as well as spatial connection, among several entities of an object. The rest of the paper is organized as follows: the second section briefly explores various studies associated with the problem of synthesizing images based on text. The implementation details are described in detail in Section 3. Section 4 highlights the key results of this research, followed by discussion in Section 5. Finally, the paper is concluded in Section 6, along with some future recommendations.

2. Background

Text-to-image synthesis is highly multimodal. Shared representation across modalities and data prediction via retrieval or synthesis are two major challenges for multimodal problems [1]. Zhu et al. [11] have exploited the capabilities of artificial intelligence (AI) and machine learning (ML) to generate images provided with the basic results. However, with the introduction of generative modeling, image generation from given text has improved drastically. Generative modeling is well suited for synthesis problems such as text-to-image synthesis [1,3,4,12,13], image to image translation [14–17], prediction of next frame in video [18], super resolution [19], etc.

Reed et al. [1] demonstrated a GAN-based architecture for translating a single text into pixels. A single deep convolutional generative adversarial network (DC-GAN) is trained conditioned on text features stored by a convolutional recurrent neural network (CRNN) in text-to-picture synthesis utilizing GANs. Feed forward inference is performed by both generators (G) and discriminators (D) based on the text feature. Both generators (G) and discriminators (D) perform feed forward inference conditioned on the text feature. Text-to-image generation using a single GAN was a huge success; however, there are some limitations as well. First, the images generated are of very low resolution and blurred. Second, less text is used for training. Finally, it was reported in their study that upon closer inspection, generated scenes are usually not coherent, which means that when the model was tested for the MS COCO [20] dataset for complex scene generation, images synthesized for composite objects were not distinguishable, and the spatial relationship between multiple objects was not fulfilled.

Another study [3] proposed a text-conditioned auxiliary classifier generative adversarial network (TAC-GAN) to improve the resolution of images synthesized from a given text. The generated images are of resolution 128×128, and the objects are more distinguishable compared to the previous text-to-image synthesizer. The Caltech UCSD birds dataset [9] is used for training. The images generated by TAC-GAN are of a high resolution, and objects are more distinguishable; however, TAC-GAN was only tested for datasets having single objects.

StackGAN [4] advocated employing many GANs layered on top of each other to create photo-realistic image synthesis. StackGAN++ [21] is a variant of StackGAN that proposes a multi-stage GAN architecture for both conditional and unconditional generative tasks. Multiple generators and discriminators are grouped in a tree-like structure in StackGAN++. Various branches of the tree produce images of the same scene with different resolutions. StackGAN++ generates images for single items successfully but fails to make photo realistic images for complex settings, as do other systems.

The attentional generative adversarial network (AttnGAN) [22] proposes picture synthesis using attention driven refinement. By paying attention to appropriate words in the text description, AttnGAN synthesizes fine-grain features at multiple sub-regions. For producing different sub-regions of the image, each attention model automatically retrieves

the most relevant word vectors. However, for complex situations, AttnGAN was unable to capture a global coherent structure.

Capsule networks have gained significant attention and success in computer vision tasks. However, their application to the domain of generating images from textual descriptions remains largely unexplored and is a promising avenue. It's interesting to note that capsule networks have primarily been utilized for detection and classification tasks in computer vision [23–25]. However, their application in text-to-image synthesis is relatively unexplored. A few authors [26–28] have explored the use of capsules with GANs; however, their models have not reported any result for text-to-image synthesis. While traditional convolutional neural networks (CNNs) and recurrent neural networks (RNNs) have been the stalwarts in this field, their inherent limitations, such as handling variations in viewpoints and object relationships, call for innovative solutions. Capsule networks, on the other hand, with their ability to capture hierarchical features and model spatial relationships, hold the potential to revolutionize text-to-image synthesis.

3. Methodology

For automatic text-to-image synthesis, we use the concept of capsule networks in an adversarial process to better model the hierarchical relationships among the entities of an object. A simple yet effective model, CapGAN is proposed to synthesize images from a given text, in which the last CNN layer at the discriminator is replaced by a state-of-the-art capsule layer to incorporate the relative spatial and orientational relationship among the various entities of an object. Photo-realistic visuals are synthesized from a given text utilizing the following four main phases in the suggested model: (1) input sentence, (2) text encoding, (3) image production and (4) image discrimination are all steps in the process. Figure 3 depicts the overall pipeline of the proposed approach. The following sections go over each stage in greater depth:
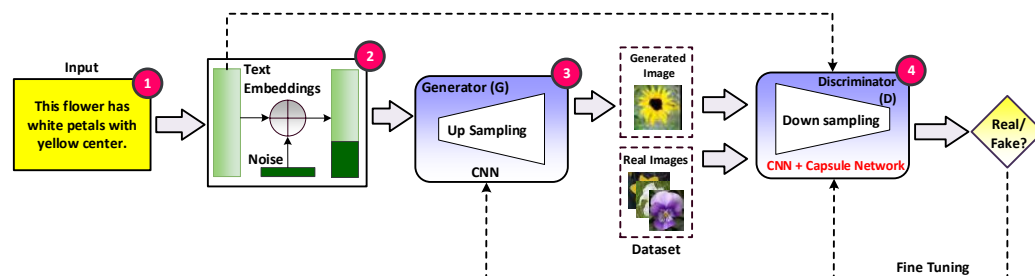

Figure 3. An illustration of the proposed CapGAN architecture for text-to-image synthesis.

3.1. Input Sentence

The input of the CapGAN architecture is a single English sentence for which an image needs to be synthesized. An example input sentence is shown in Figure 3, i.e., "The flower has white petals with yellow center". The input sentence in the next phase is encoded into a vector representation so that it can be fed to the model.

3.2. Text Encoding

The raw data from the previous step are encoded into numbers before we can use them to fit our model. For this purpose, we first use skip-thought vectors [29] for creating text embedding. Skip-thought vectors are well known neural network models that learn fixed length representations of sentences in any natural language. Figure 4 shows how skip-thought vectors transform text into image-ready vectors. The model has three parts:
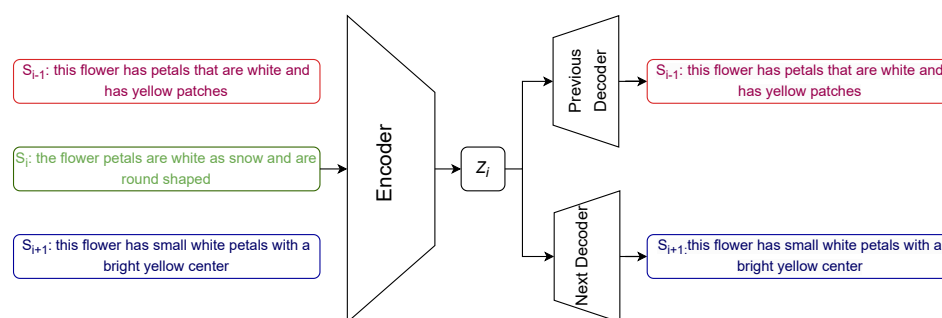
Figure 4. Skip-Thought Vector Generation: Transforming Text into Image-Ready Vectors.

1. Encoder Network: The model takes sentence i and generates a fixed length representation z using a recurrent neural network (RNN).
2. Previous Decoder Network: The model takes embedding z and tries to generate sentence $i - 1$ using RNN.
3. Next Decoder Network: The model takes embedding z and tries to generate sentence $i + 1$ using RNN.

Decoders are trained to minimize reconstruction error, which are further backpropagated to encoders for the training. Additionally, noise is added before generating the fixed length representations. The reason for corrupting, or adding noise, to the text embedding's learning process is to generate a more robust embedding space. Once trained, this trained encoder is used for generating a vector of fixed length representation, as shown in step 2 of Figure 3. This fixed length representation enables our model to replace any sentence with an equivalent vector of numbers. This caption vector is used as input for CapGAN.

3.3. Image Generation

In this section, we first give a short background of the adversarial process for automatic image synthesis and then explain the generator step of the CapGAN architecture.

3.3.1. Generative Adversarial Networks

Ian Good Fellow introduced a paradigm for estimating generative models using an adversarial process in 2014 [2], in which two models, the generator **G** and discriminator **D**, are trained. **G** generates fresh data, whereas **D** verifies that the data generated by **G** are genuine[30].

For text-to-image synthesis, GAN takes the following steps:

- **G** receives text as an input and synthesizes an image.
- **D** accepts a generated image, as well as sample images from the actual dataset, and returns the probability that the image is real, with 1 indicating a real image and 0 indicating a false image.

The core principle of GAN is depicted in Figure 5. **G** generates a sample of data from a random input **z** from **P(z)**, where **z** is a sample from the probability distribution **P(z)**. **D** receives the created data. **D** takes an input from either real data $\mathbf{x} \sim P_{data}(\mathbf{x})$ or fake data **G** and attempts to predict whether the data are real or fake. **D** uses a sigmoid function to solve a binary classification problem of real or false images and returns a value in the range of 0 to 1 [31].
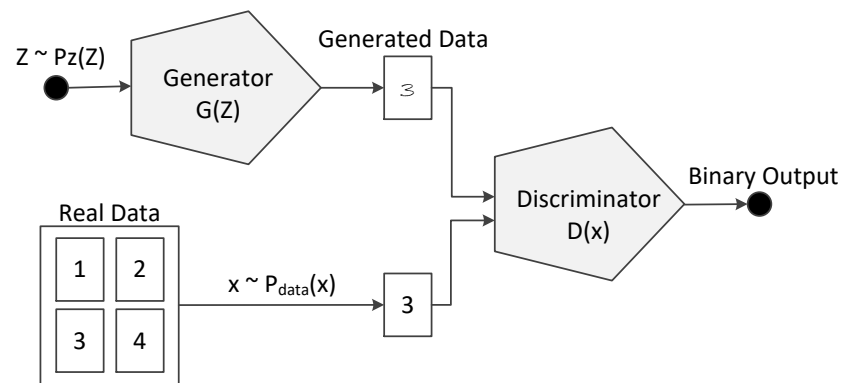
Figure 5. GAN: Idea of Generator Neural Network and Discriminator Neural Network.

GAN training takes the form of a duel between **G** and **D**. Mathematically, this can be expressed as:

$$\min_{G} \max_{D} V(D, G) \qquad (1)$$

where

$$V(D, G) = E_{x \sim \text{Pdata(x)}}[log D(x)] + E_{z \sim \text{Pz(z)}}[log(1 - D(G(z)))] \qquad (2)$$

GANs are trained on a minimax game rather than an optimization problem. The first term in function V(D,G) is the entropy, which states that the sample from real data is fed to **D** (best case scenario). **D** tries to maximize this to 1. The second term in function V(D,G) is the entropy when a sample from random distribution is fed to **D** (worst case scenario). **D** tries to minimize this to 0. Overall, **D** is trying to maximize function V. On the contrary, **G** is trying to minimize function V so that **D** cannot differentiate between real and fake. This method of training, which GAN adversarially calls the minimax game, is taken from game theory.

As the synthesis of images from extracted features is highly multimodal, the issue is not solved using deep learning. For a single input text, GANs can generate several photo realistic images. This multimodality makes GANs an ideal candidate for image synthesis problems. In consistency with this idea, the generator model of CapGAN is trained to synthesize images with basic shape and color.

3.3.2. Generator (G)

The output of the text encoding step, i.e., a caption vector, is the input of the generator network, as shown in step 3 of Figure 3. In the generator network, the caption vector of length 2400 obtained from skip-thought vectors is first compressed to acquire the text embedding of dimension 256, as shown in Figure 6. This is performed by passing the caption vector through fully connected layers, followed by LeakyReLU. The resulting text embedding is concatenated with noise, projected and reshaped into a tensor of dimension $4 \times 4 \times 1024$. This tensor is passed though the series of deconvolutions for upsampling, and as a result, a tensor of dimension $64 \times 64 \times 3$ is obtained. This tensor is a generated image from the given text, and it is fed to the discriminator for further training.
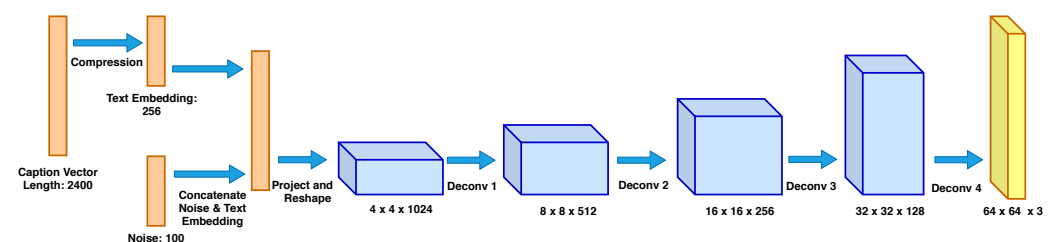

Figure 6. The generator architecture used in the proposed CapGAN model for automatic text-to-image synthesis.

3.4. Image Discrimination

3.4.1. Capsule Network

In convolution-based deep learning models such as CNN, the rotation and translation information among different pixel groups are not captured; therefore, using only convolution layers in GANs has limitations for precise image synthesis. Capsule networks [6,32,33] have recently been proposed to address this limitation of CNN. Capsules are locally invariant groups of neurons that learn to recognize the presence of visual entities in an image by encoding their properties into vector outputs [34]. In CNN, higher level features combine with low level features as a weighted sum. Nowhere in this process is there is a pose (translational and rotational) relationship between features that makes up higher level features. In a capsule network, there is a capsule corresponding to each entity in an image, which gives:

1. Probability that the entity exists.
2. Instantiation parameters of that entity.

Instantiation parameters are the properties of that entity in an image such as position, size, hue, etc. As opposed to a neuron's scalar output, a capsule outputs a vector that enables it to encapsulate all important information about the state of the feature.

Table 2 highlights the important differences between capsules and neurons. Neurons receive scalar input from low level neurons, whereas a capsule receives vector input either from low level input or from other neurons. Both the neuron and capsule perform various operations, which include transformation, weighting, summation and activation. The final output produced by the neurons at each layer is a scalar quantity, while capsules produce a vector output. The input and output vectors of capsules enable them to capture the relationship among entities of an object and make them an ideal choice for models aimed at precise image synthesis.

Table 2. Capsule vs. Neuron.

		Capsule	**Neuron**
	Input	vector(u_i)	scalar(x_i)
Operations	Linear/Affine Transformation	$\hat{u}_{ji} = W_{ij}u_i + B_j$	$a_{ji} = w_{ij}x_i + b_j$
	Weighting/Summation	$s_j = \sum_i c_{ij}\hat{u}_{ji}$	$z_j = \sum_{=1}^{3} 1.a_{ji}$
	Activation Function	$v_j = \text{squash}(s_j)$	$h_{w,b}(x) = f(z_j)$
	Output	vector(v_j)	scalar(h)

3.4.2. Discriminator (D)

The discriminator of a GAN intended for text-to-image synthesis can receive two types of input:

1. Real images with real text.
2. Synthesized/ fake images with random text.

For the proposed CapGAN model, the two types of input are shown in step 4 of Figure 3. In the CapGAN discriminator, a capsule layer is used, along with CNN layers, so that more information is retained by the vectors, thus, capturing the relationship among different entities of an object in the input image. Figure 7 shows the overall architecture of the discriminator used for the CapGAN model. Four CNN layers of stride 2 convolutions, each followed by LeakyReLU, are applied on the input image to perform downsampling. Additionally, the caption vectors of size 2400 are also transformed to text embeddings of size 256. This resultant vector, along with the output of the 4th convolutional layer, is then passed through a capsule layer, followed by the LeakyReLU and the squashing function [35]. Then, for further downsampling, the max pooling operation is applied at the output of the activation function. In the end, the discriminator resolves a binary classification problem of real or fake images using a sigmoid function and giving an output

between 0 to 1. The detailed architecture utilized for the CapGAN model is shown in Table 3.
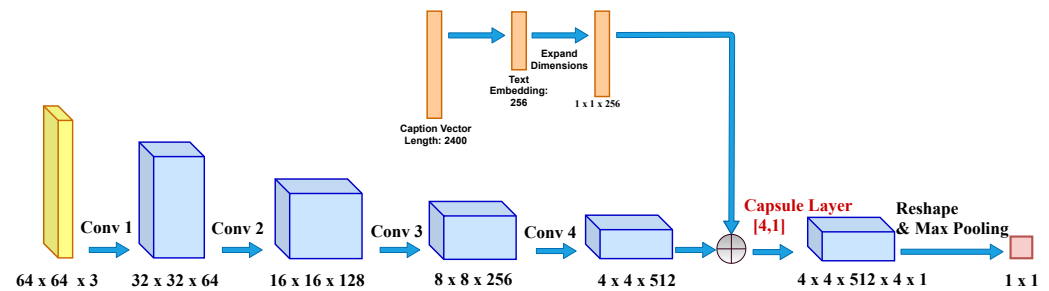

Figure 7. The discriminator architecture used in the proposed CapGAN model for automatic text-to-image synthesis.

Table 3. Details of discriminator level layers in the proposed CapGAN model for automatic text-to-image synthesis.

Layer Number	Layer Type	Input Size	Filter Size	Kernal Size	Strides	Activation Function	Output
1	Convolutional Layer	$64 \times 64 \times 3$	64	[5, 5]	[2, 2]	LeakyReLU	$32 \times 32 \times 64$
2	Convolutional Layer	$32 \times 32 \times 64$	128	[5, 5]	[2, 2]	LeakyReLU	$16 \times 16 \times 128$
3	Convolutional Layer	$16 \times 16 \times 128$	256	[5, 5]	[2, 2]	LeakyReLU	$8 \times 8 \times 256$
4	Convolutional Layer	$8 \times 8 \times 256$	512	[5, 5]	[2, 2]	LeakyReLU	$4 \times 4 \times 512$
5	Capsule Layer	$4 \times 4 \times 768$	512	[1, 1]	[1, 1]	LeakyReLU + Squashing Function	$4 \times 4 \times 512 \times 4 \times 1$

For real images, the discriminator just has to decide if an image is real. For fake images, the discriminator should distinguish two forms of errors: (1) a fake image with any text caption and (2) a real image with a mismatching text caption. In view of this, the discriminator (**D**) has to deal with the following three cases:

The first scenario is presented in Equation (3), which shows that the first real image x from the dataset, along with real text k, is given as input to **D**, while **D** computes and returns a value in the range 0 and 1, named as s_{rr}, in response to this input.

$$s_{rr} = D(x, k) \tag{3}$$

Similarly, the second scenario is depicted in Equation (4), in which **D** is given a real image as input x, along with a fake text $\hat{k}$, while **D** computes and returns a sigmoid value, named s_{rw}, in response to this input.

$$s_{rw} = D(x, \hat{k}) \tag{4}$$

Likewise, the third scenario is presented in Equation (5), in which **D** is called with a fake image $\hat{x}$ and real text k, while **D** computes and returns the sigmoid value, named s_{fr}, in response to this input.

$$s_{fr} = D(\hat{x}, k) \tag{5}$$

The three values received from Equations (3)–(5) are used to calculate the overall loss of **D**, named L_D, as shown in Equation (6).

$$L_D = log(s_{rr}) + (log(1 - s_{rw}) + log(1 - s_{fr}))/2 \tag{6}$$

The first term in Equation (6) is entropy, which is calculated when an image from the real data along with the real text is fed to **D**. **D** tries to maximize the output to 1. The second and third terms, on the other hand, show that a real image with incorrect text and a fake image with correct text are provided to **D**, respectively. **D** strives to keep this to a minimum. As a result, **D** is attempting to maximize function L_D, i.e., it is attempting to maximize the difference between its output on real and false images.

4. Results

The CapGAN architecture utilizes a capsule network for image synthesis from a given text statement to overcome the problem of global coherent structures in complex scenes. We conducted comprehensive experimentation using standardized datasets to evaluate the proposed model's performance. In the next subsections, we detail our experimental setup and also benchmark our key results.

4.1. Experimental Setup

The CapGAN model is evaluated on the Oxford-102 dataset [8], consisting of flower images, Caltech-UCSD Birds 200 [9] for bird images and ImageNet [10] for images of dogs. The detail of each dataset utilized for training and testing of CapGAN are presented in Table 4. We conducted the experiments in a ten fold cross validation setting, i.e., we conducted a total of 10 experimental rounds by performing random splits of the dataset at each round.

Table 4. Details for each dataset utilized for training and testing of CapGAN.

Dataset	Total Number of Images	Total Categories	Number of Captions per Image
Oxford-102—Flower [8]	8189	102	10
Caltech-UCSD Birds 200 [9]	6033	200	5
ImageNet Dogs [10]	4002	25	1

For each round, our CapGAN model operates using a fixed number of parameters. Table 5 lists the most important parameters, along with their values, that are used during the execution of the CapGAN model. For training, Adam [36] is used as an optimizer, and sigmoid cross entropy given logits are utilized for calculating the generator and discriminator loss. Because they are used to assess the probabilistic error in discrete classification problems where each class is independent and not mutually exclusive, we picked sigmoid cross entropy given logits. For the optimal performance, after several trials, the model is optimized for a batch size of 32, with a learning rate of 0.0002, while we run it for 100 epochs. The complete list of hyperparameters and their values can be seen in Table 5. The proposed model is trained and tested on a Tesla K40c GPU by utilizing the OpenCV, Tensorflow, Keras and CuDNN libraries.

4.2. Evaluation Metric

Evaluation metrics for supervised learning tasks are straightforward, as the problem at hand will always have a clearly defined ground truth that is always available. However, for text-to-image synthesis problems, the conventional evaluation metrics are not feasible, due to the illusive nature of the expected output, i.e., the results are highly multimodal, and no ground truth is available. Therefore, for our problem, we chose the most widely used inception score (IS) and Fréchet inception distance evaluation metrics. The details of these two metrics are given as follows:

Table 5. Hyperparameters utilized in CapGAN model.

Parameters	Value
Batch Size	32
Epochs	100
Input Image Size	$64 \times 64 \times 3$
Generated Image Size	$64 \times 64 \times 3$
Horizontal Resolution	96 dpi
Vertical Resolution	96 dpi
Bit Depth	24
Noise	100
Text Embedding	256
Caption Vector	2400
Learning Rate	0.0002
Momentum for Adam Update	0.5
Capsule Vector	[4, 1]
Generator Loss	Sigmoid Cross Entropy Given Logits
Discriminator Loss	Sigmoid Cross Entropy Given Logits

4.2.1. Inception Score

The inception score is an evaluation metric for generative models that measures "on average how different is the score distribution of synthesized images from the overall class balance" [37]. The inception score uses two criteria for measuring GAN performance:

- Saliency: Saliency indicates that objects in an image should be recognizable. Given **x** as an input, the predicted output **y** should have a high probability. In terms of image generation, given an image, an object should be recognized easily. Thus, conditional probability **p(y|x)** should be high, and as a result, the entropy is low.
- Diversity: Diversity indicates the variety of details in an image. This means, given a predicted output **y**, the marginal probability **p(y)**, should be high. This implies that for diverse images, the data distribution of **y** should be uniform, thus, resulting in high entropy.

For computing the inception score, Kullback–Leibler (KL) divergence D_{KL} is used by plugging both probabilities, i.e., the conditional probability **p(y|x)** and the marginal probability **p(y)**, as shown in Equation (8).

$$IS \approx exp(\frac{1}{N} \sum_{i=1}^{N} D_{\mathrm{KL}}(p(y|x^{(i)})||p(y)) \tag{7}$$

where N indicates the number of images generated. The intuition behind calculating the inception score is that the model should generate diverse but meaningful images. A higher value of inception score depicts that the generated images are diverse and the objects in images are highly predictable.

4.2.2. Fréchet Inception Distance

Heusel [38] proposed the Fréchet inception distance (FID), which is a variation of IS. FID is a technique for capturing the similarity between generated and real-world images. The IS calculates the quality of the synthetic images by combining the confidence of each synthesized image's conditional predictions with the marginal probability of the predicted class. Real photos are never matched with generated images in this method. The goal of

the FID score is to compare the statistics of a collection of synthetic images to the statistics of a collection of real photos in order to evaluate the synthetic images.

In order to calculate FID, the generated samples are embedded into a feature space of the inception network. The mean and covariance are estimated for both the generated data and the real data by analyzing the embedding layer as a continuous multivariate Gaussian. The Fréchet distance between these two Gaussians is then used to quantify the quality of generated samples using the following equation:

To calculate FID, the produced samples are embedded into an inception network feature space. By analyzing the embedding layer as a continuous multivariate Gaussian, the mean and covariance are computed for both the produced and real data [39]. Using the following equation, the Fréchet distance between these two Gaussians is used to measure the quality of the generated samples:

$$FID_{(r,g)} = ||\mu_r - \mu_g||_2^2 + T_r(\sum\nolimits_r + \sum\nolimits_g - 2(\sum\nolimits_r \sum\nolimits_g)^{1/2}) \tag{8}$$

The estimated mean and covariance of the real and produced data are represented by μ_r, $\sum_r$ and μ_g, $\sum_g$, respectively. This means that the lower the FID, the more realistic the resulting images are. The positive linear association between the FID score and the distorted/poorly synthesized images is depicted in Figure 8, indicating that FID is sensitive to any disruption in a created image.
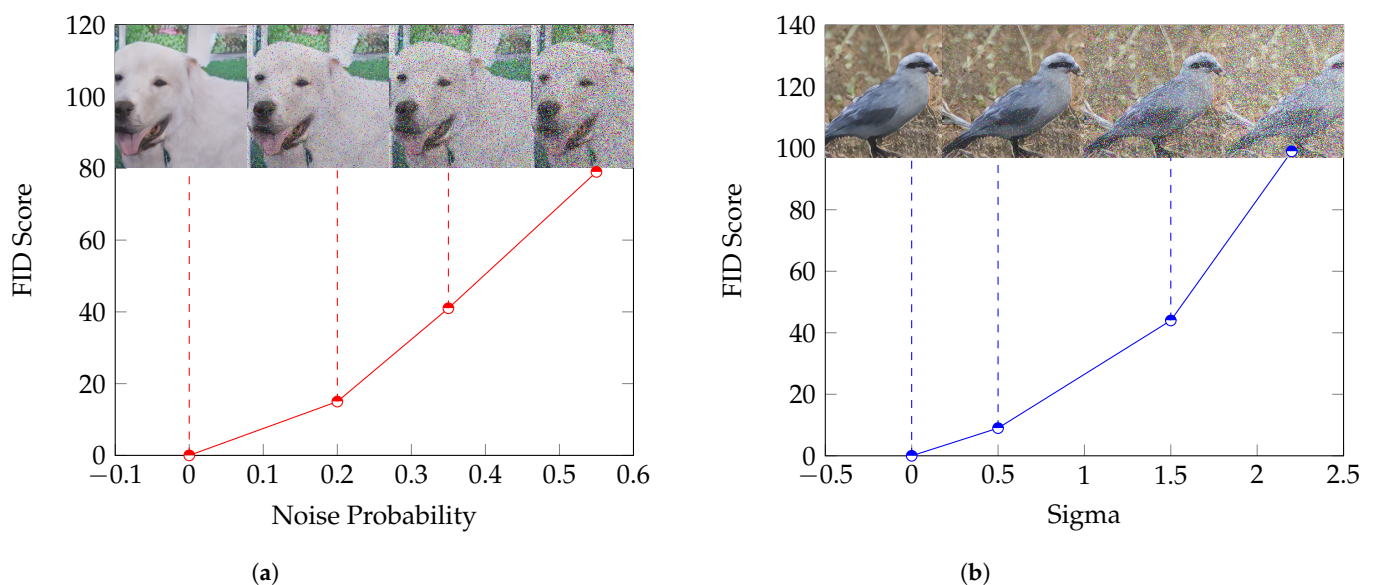
Figure 8. Rise in FID score observed as disturbance in images increases. (**a**) Salt and Pepper Noise. (**b**) Gaussian Noise.

4.3. Statistical Results

The original proposal for the inception score recommended applying the estimator (of Equation (8)) 10 times with $N = 5000$ (the number of target images). The mean and standard deviation of the obtained scores are then calculated [37]. The inception score calculated for 5000 generated images from given random captions using the CapGAN model after the 100th epoch of training remained 2.28 ± 0.627, while the inception score on the entire Oxford-102 flowers dataset [8], after the 10-fold cross validation remained 4.05 ± 0.050. The epoch-wise training results for 5000 generated images can be seen in Figure 9. The most remarkable result to emerge from these data is that as the model improves with training, the inception score raises significantly, while the standard deviation of scores initially increased and then started to decline. This indicates that the diversity of generated images and their predictability increases considerably with more training. We stopped our model training at the 100th epoch as the model was tending towards overfitting after this iteration. The calculated IS and FID for various complete datasets using CapGAN are listed in Table 6.
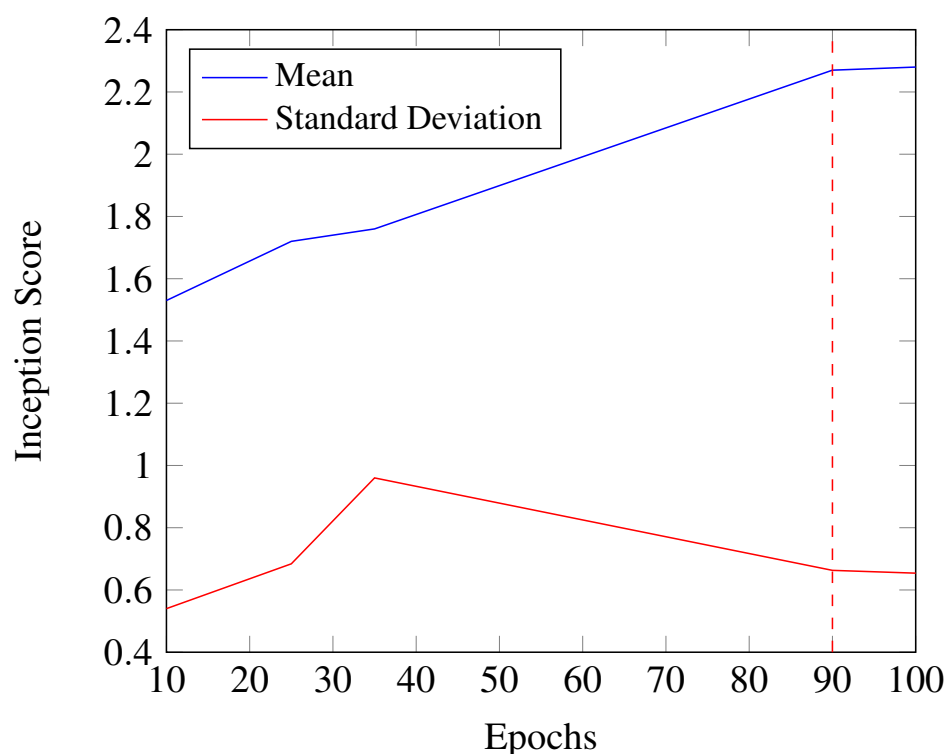
Figure 9. The inception score plotted against epochs during training.

Table 6. IS and FID score calculated using CapGAN.

Dataset	IS	FID
Oxford-102—Flower	4.05 ± 0.050	47.38
Caltech-UCSD Birds 200	4.61 ± 0.1	14.98
ImageNet Dogs	13.11 ± 0.407	38.18

4.4. Visual Results

Another interesting aspect of looking at the results is through the visual inspection of the synthesized images. For this purpose, we present the images generated using CapGAN in Table 7. For each input caption, we report ten images of size 64×64 generated after the 25th and 100th epochs using CapGAN. As an example, for the first caption, i.e., *This flower has petals that are yellow and has black stamen.*, the model instantly learns the trivial details such as yellow color, as can be seen in the 25th epoch images, but the global coherent structure, e.g., petals on the flower are not learned well at this stage. As the learning continues, the *sigmoid cross entropy given logits* loss at the generator and discriminator reduces significantly, which ultimately improves the inception score. Figure 10 shows losses for G and D calculated for GAN and CapGAN while training. After the 100th epoch, when the model is fully trained, the model learns the coherent structure details described in the input sentence, as can be seen in the second row of the first caption output in Table 7. Similarly, the same ability of the model can be observed for all other captions reported in Table 7. Moreover, samples of the various dog images generated by CapGAN model trained with 100 epochs are listed in Figure 11. These results offer compelling evidence about the ability of the CapGAN model to learn the spatial relationships among different entities of an object in an image.

(**a**) CapGAN Generator

(**b**) CapGAN Discriminator

(**c**) GAN Generator

(**d**) GAN Discriminator
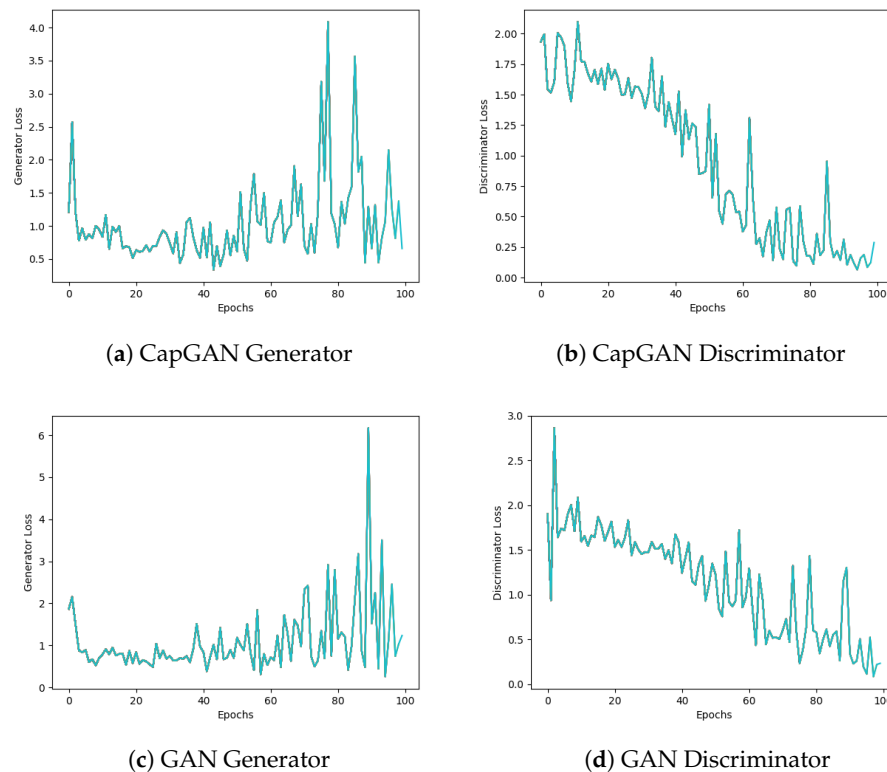
Figure 10. (**a**,**b**) shows losses for **G** and **D** for CapGAN, respectively. For **D**, the loss decreases as the epochs increase. However, the loss of **G** starts increasing after the 60th epoch which indicates that **D** became too strong relative to the **G**. Beyond this point, **G** finds it almost impossible to fool **D**. When **D** loss decreases to a small value (i.e., 0.1 to 0.2) and **G** loss increases to a high value (i.e., 2 to 3), it means that model is trained, as **G** cannot be further improved. (**c**,**d**) are losses for **G** and **D** of GAN: To calculate the loss, all layers are kept as convolutional layers. In comparison, **D** loss for CapGAN is less than GAN.

Figure 11. Sample of the dog images generated by the CapGAN model trained on the ImageNet dataset.

4.5. Comparative Results

CapGAN is compared to the earlier state-of-the-art models for text-to-image synthesis to further highlight the usefulness of the proposed model. Our model is compared against GAN [1], StackGAN [4], StackGAN++ [21] and TAC-GAN [3] architectures. All these methods utilize GAN architecture as the backbone for translating a single sentence directly into pixels. A single deep convolutional generative adversarial network (DC-GAN) is trained and conditioned on text features encoded by a convolutional recurrent neural network (CRNN) in text-to-image synthesis utilizing GANs . Feed forward inference is performed by both the generator (G) and the discriminator (D) based on the text feature. Among all these models, the text-conditioned auxiliary classifier generative adversarial network

(TAC-GAN) is specifically designed to improve the resolution of images synthesized for complex scenes from a given text. In TAC-GAN, the generator is a neural network made up of a series of transposed convolutional layers, while the discriminator is a network that takes an input image and passes it through a number of convolutional layers to determine if the resulting image is real or fake.

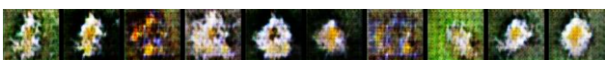**Table 7.** Examples of early and final stage images generated from various input texts using CapGAN.
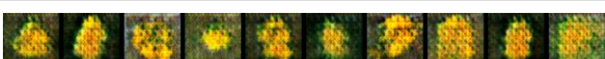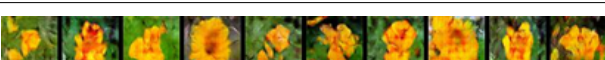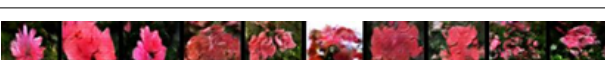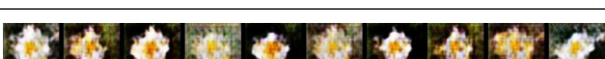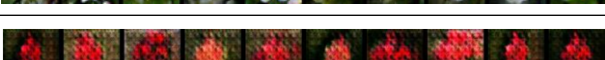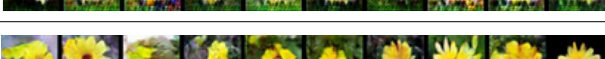
Input Text	Epoch	Examples of Generated Images
This flower has petals that are yellow and has black stamen.	25	
	100	
The pretty flower has a lot of short blue petals.	25	
	100	
The flower has petals that are yellow with orange spots.	25	
	100	
This flower is pink in color, with petals that are wavy and bunched together.	25	
	100	
This flower has petals that are white with a yellow center.	25	
	100	
This flower is red and tan in color, with petals that are spotted.	25	
	100	
A flower with light yellow petals and yellow pistils in the center.	25	
	100	

Table 8 lists the ISs and FIDs calculated using different models for the Oxford-102 flowers dataset [8] Caltech-UCSD 200 [9] and ImageNet [10] for images of dogs and evidently shows that CapGAN achieves the highest inception score and lowest Fréchet inception distance and outperforms the previous models. In comparison to the other models, a higher inception score indicates that CapGAN's images are more recognized, meaningful and have a greater diversity of information. The lower FID scores suggest that generated images are less distorted and closer to real-world images.

Table 8. The inception score (IS) and and the Fréchet inception score (FID) score calculated using CapGan.

Model	Dataset	Oxford-102—Flower	Caltech-UCSD Birds 200	ImageNet Dogs
GAN [1]	IS ↑	2.66 ± 0.03	2.78 ± 0.1	6.81 ± 0.76
	FID ↓	76.98	53.89	98.01
StackGAN [4]	IS ↑	3.20 ± 0.01	3.70 ± 0.04	8.84 ± 0.08
	FID ↓	55.28	51.89	89.21
StackGAN++ [21]	IS ↑	3.26 ± 0.01	4.04 ± 0.5	9.55 ± 0.11
	FID ↓	48.68	15.30	44.54
TAC-GAN [3]	IS ↑	3.45 ± 0.05	-	-
	FID ↓	-	-	-
CapGAN	IS ↑	**4.05 ± 0.050**	**4.12 ± 0.023**	**11.35 ± 0.11**
	FID ↓	**44.38**	**11.89**	**34.36**

5. Discussion

In complex scenes, objects are composed of multiple entities that are interlinked to form a whole part; however, the color(s) and basic shape of each entity in the scene can be fully viewed and determined separately. For complex scenes, we proposed and evaluated a new model called CapGAN that utilizes a capsule network for image synthesis from a given text statement to overcome the problem of global coherent structures in complex scenes. Our model uses skip-thought vectors as text encoders to construct highly generic fixed-length sentence representations from a single text statement as input. This encoded vector is utilized as input for image synthesis, utilizing an adversarial approach in which two models, generator (G) and discriminator (D), are trained simultaneously. Our model is conceptually unique in that it integrates capsules at the discriminator level to allow it to grasp the orientational and relative spatial relationships between different elements of an object.

To better understand the effectiveness of using a capsule layer at the discriminator lever, we compare the images generated using GAN (without capsules) with images synthesized using the CapGAN model. Table 9 shows images generated using GAN and CapGAN. For GAN, all layers are kept as conventional convolutional layers at the discriminator level. However, for CapGAN, capsule layers are integrated at the discriminator level. From Table 9, it is clear that images generated using the capsule layer at the discriminator level are visually more appealing than the images generated using conventional layers. In a similar vein, it is worth noting that the saliency (i.e., the probability of object presence in the synthesized images) and the diversity (i.e., the variations in the synthesized images) are way better in the CapGAN model than the GAN model.

5.1. Multimodality Preservance

The problem of generating images from text descriptions is highly multimodal. This means that there can be multiple correct answers for a single input sentence. When it comes to image synthesis from text, multimodality implies that there are numerous possible pixel configurations that can accurately depict the same description. The CapGAN model also preserves the multimodality. To ensure multimodality, in Table 10, the images generated randomly from a given text using CapGAN are compared with images from the dataset. The images in both columns are different from each other; however, it can clearly be seen that they are all correct in the visual illustration of the given input text, i.e., the entities, color(s) and shape of each entity mentioned in the input text is present in the generated image. As an example, for the first sentence, i.e., *"This flower has a white petal with a yellow center."*, both the dataset and the generated image have a flower with white petals and a yellow center. Similarly, for the text: *This particular bird has a belly that is gray and white*, the model has generated a bird with a gray and white color; however, the close inspection of a

ground truth indicates that the bird has also a yellow beak. Nevertheless, this information was not available in text, thus, in the generated image, the bird has a white beak.

Table 9. Comparison of images generated from a given text using the GAN and CapGAN models.

Text	Model	Sample of Images Generated
This flower has long yellow petals that are curved down and a black center with black anthers on it.	GAN	
	CapGAN	
This flower is white and yellow in color, and has petals that are yellow near the center.	GAN	
	CapGAN	
This flower is pink in color, and has petals that are oddly shaped and vertically layered.	GAN	
	CapGAN	
This is a bird with grey wings, a white neck and a black beak.	GAN	
	CapGAN	
This bird is red in color, with black wings.	GAN	
	CapGAN	
This particular bird has a belly that is gray and yellow.	GAN	
	CapGAN	

5.2. Synthesis of Global Coherent Structures

It is also of interest to see the correlation between coherent structures in complex scenes, and the ability of CapGAN to synthesize them. For this purpose, we integrated the capsule networks at the discriminator level in the CapGAN model. The capsule networks extract the geometric information of an object in an image in the form of vectors and use it for inverse rendering. Therefore, in contrast to the conventional deep networks, a capsule network easily identifies the spatial associations among an object's several entities in a scene.

The power of capsule networks appears to be well-substantiated by the results produced using the CapGAN model. The images generated by CapGAN evidently show that they are closer to the given text and have more relative spatial and orientational association between objects, as well as group of pixels, in comparison to images generated using conventional networks. For instance, the images generated by CapGAN using the first text shown in Table 9, the flowers have *long petals*, they are more *curved down* and have a proper *black center* compared to images generated by GAN. Moreover, the majority of the images generated by CapGAN are close to realistic images.

Table 10. Images generated vs. images from dataset.

Text	Ground Truth	Generated Images Using CapGAN
This flower has a white petal with a yellow center.		
This flower has red petals with white center.		
This flower has a yellow petal with orange spots.		
This flower has pink petals with a pink center.		
This bird is yellow and black in color, with a long black beak.		
This particular bird has a belly that is gray and white.		
This is a brown and beige bird and brown on the crown		
White Shih-Tzu		

On the other hand, many images generated by GAN are far from reality if we compare them with the expected output of the input text. For example, in the second text, as shown in Table 9, white and yellow colors are merged together in images generated by GAN, but for CapGAN, the transition from the yellow center to white leaves is much more smooth and close to reality. Likewise, in images generated from the third text using CapGAN, the petals are more pink, vertically layered and connected, compared to petals in images synthesized by GAN. In many GAN images, the connection between the petals and various parts of the flowers are missing, while they are preserved and well-synthesized using CapGAN. Thus, all these findings correlate favorably with our argument and further support the idea that the CapGAN model outshines in capturing the color(s), basic shape of each entity in the scene, as well as the spatial relationships between objects in complex scenes.

6. Conclusions

We proposed and tested a model called CapGAN for generating images from a given text statement in this paper. The proposed model is based on an adversarial process in which two models, generator (G) and discriminator (D), are trained simultaneously. The convolutional layers are replaced by capsule layers in CapGAN's discriminator stage. The capsules outperform traditional convolutional neural networks because they incorporate orientation and relative spatial interactions between various objects. The suggested CapGAN model's usefulness is convincingly demonstrated by the experimental findings, which is especially important for generating images for complicated scenarios. For the image synthesis problem, the suggested model outperforms the existing state-of-the-art models. In future, the model developed in this research can be scaled up to generate higher resolution images. Since the GANs capability is limited by the generator's potential, in

future traditional deconvolutional neural network at the generator level can be replaced by anti-capsule networks for better results. Furthermore, many approaches have used multistage GAN architecture for increasing the image resolution, where the output obtained in the first phase is alternatively passed to the next phase. It is believed that results from CapGAN can be further improved from using such multistage architectures.

Author Contributions: H.U.R. and A.B. supplied the domain knowledge and framed the problem, while M.O. devised the idea of using a capsule network in conjunction with a convolutional neural network for text-to-image synthesis. The experiment code was written by M.O. and H.U.R. Both O.B.S. and G.P. assisted with the execution of experiments, as well as the analysis and preparation of figures. M.A. assisted with the execution of experiments, analysis, preparation of figures and visualization of the experimental results. In the production and revision of the work, all authors contributed equally. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: The data presented in this study are publicly available at the following articles: (1) Oxford-102 dataset [8], consisting of flower images. (2) Caltech-UCSD Birds 200 dataset [9] for bird images, and (3) Dog image dataset from ImageNet [10] for images of dogs.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Reed, S.; Akata, Z.; Yan, X.; Logeswaran, L.; Schiele, B.; Lee, H. Generative Adversarial Text-to-Image Synthesis. In Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 20–22 June 2016.
2. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. In Proceedings of the Annual Conference on Neural Information Processing Systems 2014, Montreal, QC, Canada, 8–13 December 2014.
3. Dash, A.; Gamboa, J.C.B.; Ahmed, S.; Liwicki, M.; Afzal, M.Z. TAC-GAN-text conditioned auxiliary classifier generative adversarial network. *arXiv* **2017**, arXiv:1703.06412.
4. Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Huang, X.; Wang, X.; Metaxas, D. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In Proceedings of the IEEE International Conference on Computer Vision 2017, Venice, Italy, 22–29 October 2017.
5. Dong, H.; Zhang, J.; McIlwraith, D.; Guo, Y. I2T2I: Learning text to image synthesis with textual data augmentation. In Proceedings of the 2017 IEEE International Conference on Image Processing (ICIP), Beijing, China, 17–20 September 2017.
6. Sabour, S.; Frosst, N.; Hinton, G.E. Dynamic routing between capsules. In Proceedings of the 2017 Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.
7. Dai, A.M.; Le, Q.V. Semi-supervised sequence learning. In Proceedings of the 2015 Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–10 December 2015.
8. Nilsback, M.E.; Zisserman, A. Automated flower classification over a large number of classes. In Proceedings of the 2008 Sixth Indian Conference on Computer Vision, Graphics & Image Processing, Bhubaneswar, India, 16–19 December 2008.
9. Welinder, P.; Branson, S.; Mita, T.; Wah, C.; Schroff, F.; Belongie, S.; Perona, P. *Caltech-UCSD Birds 200*; Technical Report CNS-TR-2010-001; California Institute of Technology: Pasadena, CA, USA, 2010.
10. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet large scale visual recognition challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
11. Zhu, X.; Goldberg, A.B.; Eldawy, M.; Dyer, C.R.; Strock, B. A text-to-picture synthesis system for augmenting communication. In Proceedings of the AAAI 2007, Vancouver, BC, Canada, 22–26 July 2007; Volume 7, pp. 1590–1595.
12. Zhang, Z.; Xie, Y.; Yang, L. Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018.
13. Chen, Q.; Koltun, V. Photographic Image Synthesis with Cascaded Refinement Networks. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
14. Sangkloy, P.; Lu, J.; Fang, C.; Yu, F.; Hays, J. Scribbler: Controlling Deep Image Synthesis With Sketch and Color. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
15. Nie, D.; Trullo, R.; Lian, J.; Petitjean, C.; Ruan, S.; Wang, Q.; Shen, D. Medical image synthesis with context-aware generative adversarial networks. In Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention, Quebec City, QC, Canada, 11–13 September 2017.

16. Dong, H.; Yu, S.; Wu, C.; Guo, Y. Semantic image synthesis via adversarial learning. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017.

17. Wang, T.C.; Liu, M.Y.; Zhu, J.Y.; Tao, A.; Kautz, J.; Catanzaro, B. High-Resolution Image Synthesis and Semantic Manipulation With Conditional GANs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.

18. Liang, X.; Lee, L.; Dai, W.; Xing, E.P. Dual motion GAN for future-flow embedded video prediction. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.

19. Ledig, C.; Theis, L.; Huszár, F.; Caballero, J.; Cunningham, A.; Acosta, A.; Aitken, A.P.; Tejani, A.; Totz, J.; Wang, Z.; et al. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. In Proceedings of the CVPR 2017, Honolulu, HI, USA, 21–26 July 2017.

20. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft coco: Common objects in context. In Proceedings of the European Conference on Computer Vision, Zurich, Switzerland, 6–12 September 2014.

21. Zhang, H.; Xu, T.; Li, H.; Zhang, S.; Wang, X.; Huang, X.; Metaxas, D. StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *41*, 1947–1962. [CrossRef] [PubMed]

22. Xu, T.; Zhang, P.; Huang, Q.; Zhang, H.; Gan, Z.; Huang, X.; He, X. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *arXiv* **2017**, arXiv:1711.10485.

23. Afshar, P.; Mohammadi, A.; Plataniotis, K.N. Brain Tumor Type Classification via Capsule Networks. In Proceedings of the 2018 25th IEEE International Conference on Image Processing (ICIP), Athens, Greece, 7–10 October 2018.

24. Lukic, V.; Brüggen, M.; Mingo, B.; Croston, J.H.; Kasieczka, G.; Best, P.N. Morphological classification of radio galaxies: Capsule networks versus convolutional neural networks. *Mon. Not. R. Astron. Soc.* **2019**, *487*, 1729–1744. [CrossRef]

25. Hilton, C.; Parameswaran, S.; Dotter, M.; Ward, C.M.; Harguess, J. Classification of maritime vessels using capsule networks. In *Geospatial Informatics IX*; SPIE: Bellingham, WA, USA, 2019; Volume 10992, pp. 87–93

26. Bass, C.; Dai, T.; Billot, B.; Arulkumaran, K.; Creswell, A.; Clopath, C.; De Paola, V.; Bharath, A.A. Image synthesis with a convolutional capsule generative adversarial network. In Proceedings of the International Conference on Medical Imaging with Deep Learning, London, UK, 8–10 July 2019.

27. Jaiswal, A.; AbdAlmageed, W.; Wu, Y.; Natarajan, P. Capsulegan: Generative adversarial capsule network. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.

28. Upadhyay, Y.; Schrater, P. Generative adversarial network architectures for image synthesis using capsule networks. *arXiv* **2018**, arXiv:1806.03796.

29. Kiros, R.; Zhu, Y.; Salakhutdinov, R.R.; Zemel, R.; Urtasun, R.; Torralba, A.; Fidler, S. Skip-thought vectors. In Proceedings of the 2015 Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–10 December 2015.

30. Xie, J.; Yu, F.R.; Huang, T.; Xie, R.; Liu, J.; Wang, C.; Liu, Y. A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges. *IEEE Commun. Surv. Tutorials* **2019**, *21*, 393–430. [CrossRef]

31. Nguyen, T.; Vu, P.; Pham, H.; Nguyen, T. Deep learning UI design patterns of mobile apps. In Proceedings of the 2018 IEEE/ACM 40th International Conference on Software Engineering: New Ideas and Emerging Technologies Results (ICSE-NIER), Gothenburg, Sweden, 27 May–3 June 2018.

32. Hinton, G.E.; Krizhevsky, A.; Wang, S.D. Transforming auto-encoders. In Proceedings of the International Conference on Artificial Neural Networks, Espoo, Finland, 14–17 June 2011.

33. Hinton, G.E.; Sabour, S.; Frosst, N. Matrix capsules with EM routing. In Proceedings of the 6th International Conference on Learning Representations ICLR 2018, Vancouver, BC, Canada, 30 April–3 May 2018.

34. Li, S.; Ren, X.; Yang, L. Fully CapsNet for Semantic Segmentation: First Chinese Conference. In Proceedings of the PRCV 2018, Guangzhou, China, 23–26 November 2018.

35. Nair, P.; Doshi, R.; Keselj, S. Pushing the limits of capsule networks. *arXiv* **2018**, arXiv:2103.08074.

36. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. In Proceedings of the 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, 7–9 May 2015.

37. Barratt, S.; Sharma, R.K. A Note on the Inception Score. *arXiv* **2018**, arXiv:1801.01973.

38. Heusel, M.; Ramsauer, H.; Unterthiner, T.; Nessler, B.; Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Proceedings of the 2017 Conference on Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017.

39. Borji, A. Pros and cons of gan evaluation measures. *Comput. Vis. Image Underst.* **2019**, *179*, 41–65. [CrossRef]