Abstract

Text-to-image generation is an active area of research, where many possible applications can be applied. Most of the existing research projects are primarily focused on generating a single image from available information. The whole process is just one step. The extension beyond this one-step generation is a system that generates an image iteratively, conditioned on ongoing linguistic input or feedback. This is more challenging than one-step generation tasks, as such a system must understand the contents of its generated images with respect to the feedback history of the previous images, the current feedback, as well as the interactions among concepts present in the feedback history. Here, we present a recurrent image generation model which takes into account both the generated output up to the current step as well as all past instructions for generation. We show that our model is able to generate the background (blank canvas), add new objects, and apply simple transformations to existing objects. Starting from the empty or blank canvas we do the generation of images iteratively where the teller gives the instructions and the drawer performs the actions based on the linguistic inputs.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Communication among people has always played a vital role right from the early days of human civilization and has greatly contributed and continues to contribute to the progress of the society. We always chased the ability to perfectly articulate our opinions, stories, emotions.. It is the research in neuroscience for the art of presentation which tells us that visual depiction of data/information is more pleasing to the brain and easier to process and remember. Vision is the medium through which we mostly interact and experience most of the time. Machines that can understand the world and generate new images and videos have wide applications in pioneering interactive education, multimedia creation, and creative arts.

A system that can understand and interpret the human language and give visual feedback of its interpretation has been an enthralling research topic and is being pursued over the years. It is such a system that enhances the human ability to articulate with a greater meaning. Such a system is necessary in modern times where diseases like Aphasia are on the raise and stand as a barrier for communication.

We aim to build a model that can iteratively draw images based on natural human speech. It should be able to modify the previously generated images based on the input text at the current time step and the history of previous instructions. This is possible with the principles of advanced Computer Vision intersected with Natural Language Processing. We take a base paper that deals with this problem of Iterative Image-to-Text conversion.

Chapter 2

Problem Definition

As mentioned before in the introduction section, there is always a better way to communicate thoughts. There are people who are deaf, mentally challenged, and suffering from dyslexia. Visual representation of data, information or statistics will make comprehension easier for people with such disorders. Also, language is an obstacle to communicate with people across the world. Though there are neural machine translation models that convert text from one language to another, they are only limited to the popular languages across the world. The data to train a translation machine for the regional languages of an area are insufficient. Therefore it is necessary to find a way to convert the human language with complex linguistic rules into visuals. Images are a universal way of representation of data.

Continuous generation of images will revolutionize the multimedia industry and will pave the way for the rapid creation of content for professional artists or educators. It will help them exactly elucidate what they have in their minds. There has been some work on converting text to image. However, these models are not interactive. Meaning, these cannot take feedback on the generated image from the user based on the user's visual understanding of what the model has drawn. We believe this addition of interactiveness will further help the user to steer the model to draw what the user actually needs.

We start by exploring the dataset that is required for this task. Having just images and corresponding text is not sufficient for this problem. We need a dataset where each sample in the dataset is a consecutive sequence of image-label pairs with each image-label pair directly dependent on the previous image-label pairs. Such dataset is the iterative version of CLEVR

(A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning) which is the i-CLEVR dataset. The i-CLEVR dataset consists of 10,000 sequences, each sequence having 5 images with their corresponding text. That makes it a total of 50,000 images and instructions.

We model an adversarial model consisting of a Generator and a Discriminator. The Generator's task is to take in a noise image (usually Gaussian or Normal) and the text from the current time step encoded by a Gated Recurrent Network and previous instructions encoded by an Bi-Directional Gated Recurrent Network. The Discriminator's job is to take in the ground truth image for the text and then calculate a measure called rmsim, the similarity between the generated image and ground truth image. The Discriminator will then give a loss value based on the rmsim value. This loss value is later fed into the Generator and the Generator adjusts its weights so that the loss value at the Discriminator will increase. We try to minimize the loss of Generator and maximise the loss of the Discriminator so that the Generator will be able to bypass the Discriminator test. Thus paving way to building a Recurrent GAN model which can draw upon the previous generated images.

Chapter 3

Related Work

A lot of research across the world has been conducted to build a model that takes user input as text, interprets it and then converts that information from the text embeddings space to 2-D image space where the image will closely resemble what the input text describes. Adversarial models such as StackGAN, StackGAN++, and AttnGAN have been able to generate high-resolution images capturing the semantic meaning from the given input text. The drawback in these models is their inability to take in feedback and generate new images from a new instruction and previously generated image.

GANs [1] represent a powerful family of generative models whose benefits and strengths extend to conditional image generation. Several approaches for conditioning exist, such as conditioning both the generator and discriminator on labels [2], as well as training an auxiliary classifier as part of the discriminator [3]. Closer to GeNeVA text-based conditioning, Reed et al. [4] generate images conditioned on the provided captions.

Zhang et al. [5] proposed a two-stage model called StackGAN, where the first stage generated low resolution images conditioned on the caption, and the second stage generated a higher resolution image conditioned on the previous image and the caption. Hong et al.[6] proposed a three-step generation process where they use external segmentation and bounding box annotation for MS COCO to first generate bounding boxes, then a mask for the object, and then the final image.

Building upon StackGAN,AttnGAN [7] introduced an attentional generator network that enabled the generator to synthesize different spatial lo-

cations in the image, conditioned on an attention mechanism over words in the caption. It also introduced an image-text similarity module which encouraged generating images more relevant to the provided caption. Departing from purely caption data, Sharma et al. [8]proposed a non-iterative model called ChatPainter that generates images using dialogue data. ChatPainter conditions on captions from MS COCO and a Recurrent Neural-Network (RNN)-encoded dialogue relevant to the caption (obtained from the Visual Dialog (VisDial) [9] dataset)to generate images. The authors showed that the question answering-based dialogue captured richer information about the image than just the caption, which enabled Chat-Painter to generate superior images compared to using captions alone. Since the VisDial dialogues were collected separately from the MS COCO dataset, there are no intermediate incremental images for each turn of the dialogue. The model, thus, only reads the entire dialogue and generates a single final image, so this setup diverges from a real-life sketch artist scenario where the artist has to keep making changes to the current sketch based on feedback.

There has also been recent work in performing recurrent image generation outside of text-to-image generation tasks.Yang et al. [10] perform unsupervised image generation in recursive steps, first generating a background, subsequently conditioning on it to generate the foreground and the mask,and finally using an affine transformation to combine the foreground and background.

Lin et al. [11] tackle the image compositing task of placing a foreground object on a background image in a natural location. However, this approach is limited to fixed object templates, and instead of generating images directly, the model recursively generates parameters of transformations to continue applying to an object template until the image is close enough to natural image manifold. The approach also does not modify existing objects in the image. Both of these approaches aim to generate a single final image without incorporating any external feedback.

Chapter 4

Proposed System

The proposed system is a Recurrent Generative Adversarial Network. We define a task which has to be solved by the model. This task has two players. The teller and the Drawer. The Teller tells the instructions to the Drawer and the Drawer tries to draw what the Teller has described. First, an empty canvas will be given to the Drawer and then, iteratively Teller and Drawer will make their actions leading way to continuous Text-to-Image. Here the Teller is modeled as the input text given to the GAN and Drawer is modeled as the GAN.

There are other components like an Image Encoder and an Sentence Encoder which helps the flow of data from the Teller to Drawer and back to Teller. The Image Encoder consists of two blocks: Residual up block and Residual down block. Residual down block takes an input image and extracts the features from it, thus down-sampling the image. Residual up block takes in a lower dimensional image and up-samples it into the size of the original image.

Our Generator is an Image Encoder with Residual up block. It takes in a noise and conditional augmented output from a Gated Recurrent Network to which current input text and previous context-aware texts are given as inputs. Before inputting the current input text and previous texts are given as inputs, they need to be converted into context-aware vectors for the Gated Recurrent Unit to understand what the text and images describe. So the current input text is encoded with the help of a Sentence Encoder which is a Bi-directional RNN using GLoVe embeddings as input embeddings. We fuse together the image pooled features extracted from the Image Encoder to which the base background image is given as input and the encoded sentence.

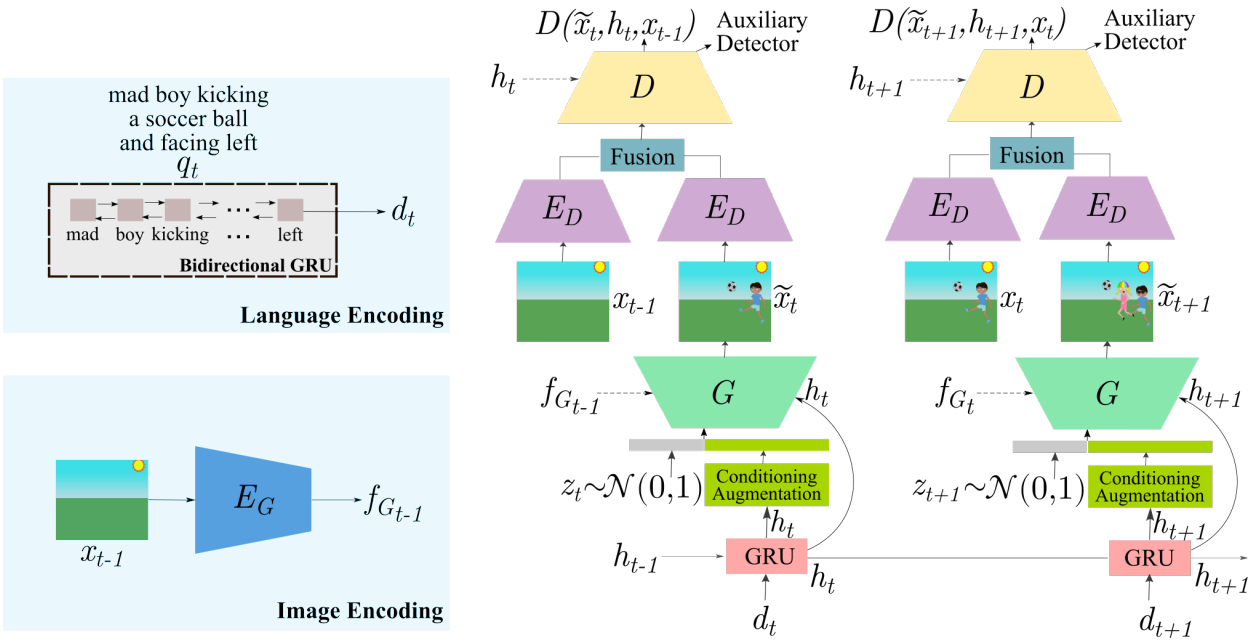

Figure 4.1: Model Architecture

The Discriminator takes in the generated image and the ground truth image of the previous time step. It also takes the context-aware vector which contains the semantic meaning of the history of previous instructions also the previously generated images. This helps the Discriminator to see how relevant the generated image is with respect to the given instruction and previous image.

It will give us a measure of how good the modifications are. If we simply try to compare the generated image and the ground truth image of the current step using a generic similarity function like Fréchet Inception Distance (FID), it will only tell us how realistic the generated images looks but will be unable to tell if the modifications made by the generator are relevant to the input instruction given by the user or not.

An auxiliary objective is added to the Discriminator, which is an object detection and localization ability. This helps the Discriminator compare the objects in the generated image from the previous time step and objects in the generated image from the current time step, thus telling us if the number of objects or type of objects have been placed or not.The object localiser estimated the position of a object in a image. It tells us if the generator was able to capture the relative position information of objects from the input

text and place it accordingly in the fake image efficiently or not. This object detector and localizer is based on the Inception-V3 model.

To compare the arrangement of objects qualitatively, we use the above object detector/localizer to determine the type and position of objects in the ground truth and the generated image. We estimate a scene graph for each image, in which the detected objects and the image center are the vertices. The directed edges are given by the left-right and front-back relations between the vertices. To compute a relational similarity metric on scene graphs, we determine how many of the ground truth relations are present in the generated image.

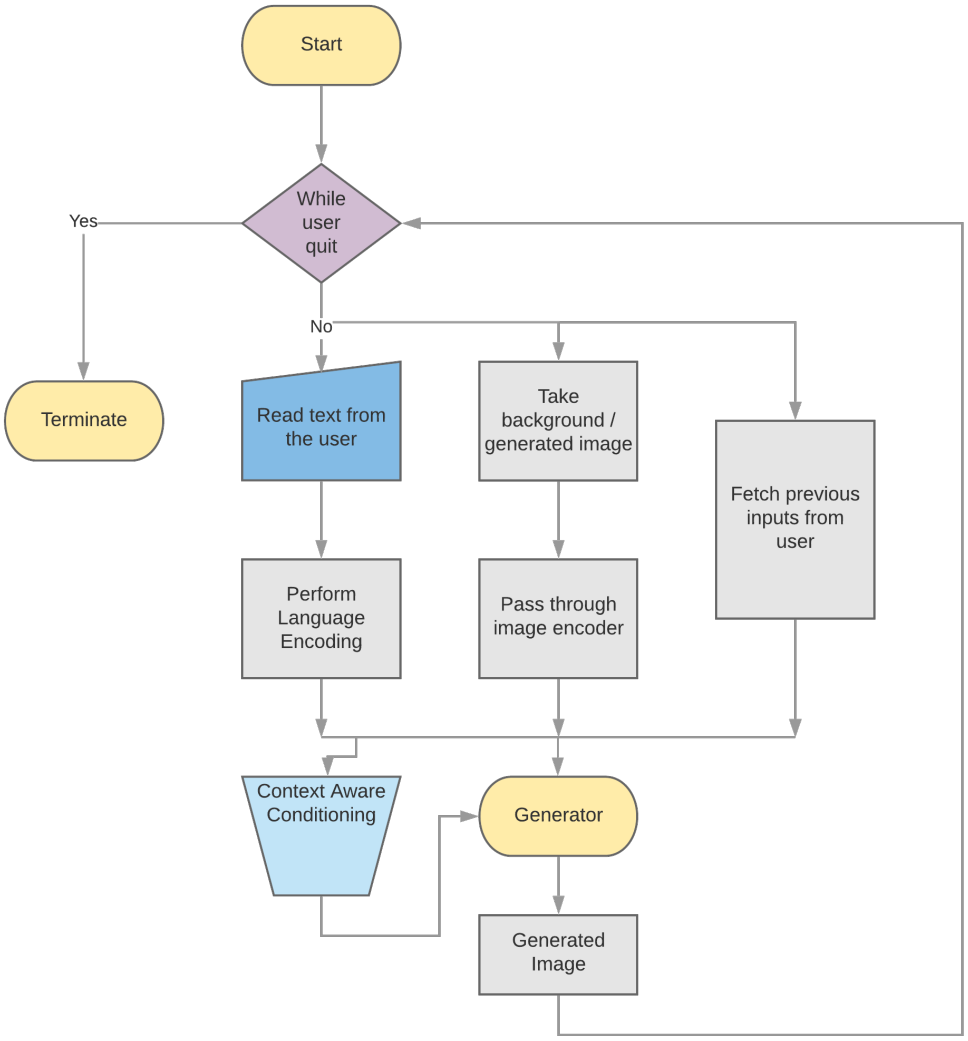

Figure 4.2: Data Flow for Testing phase

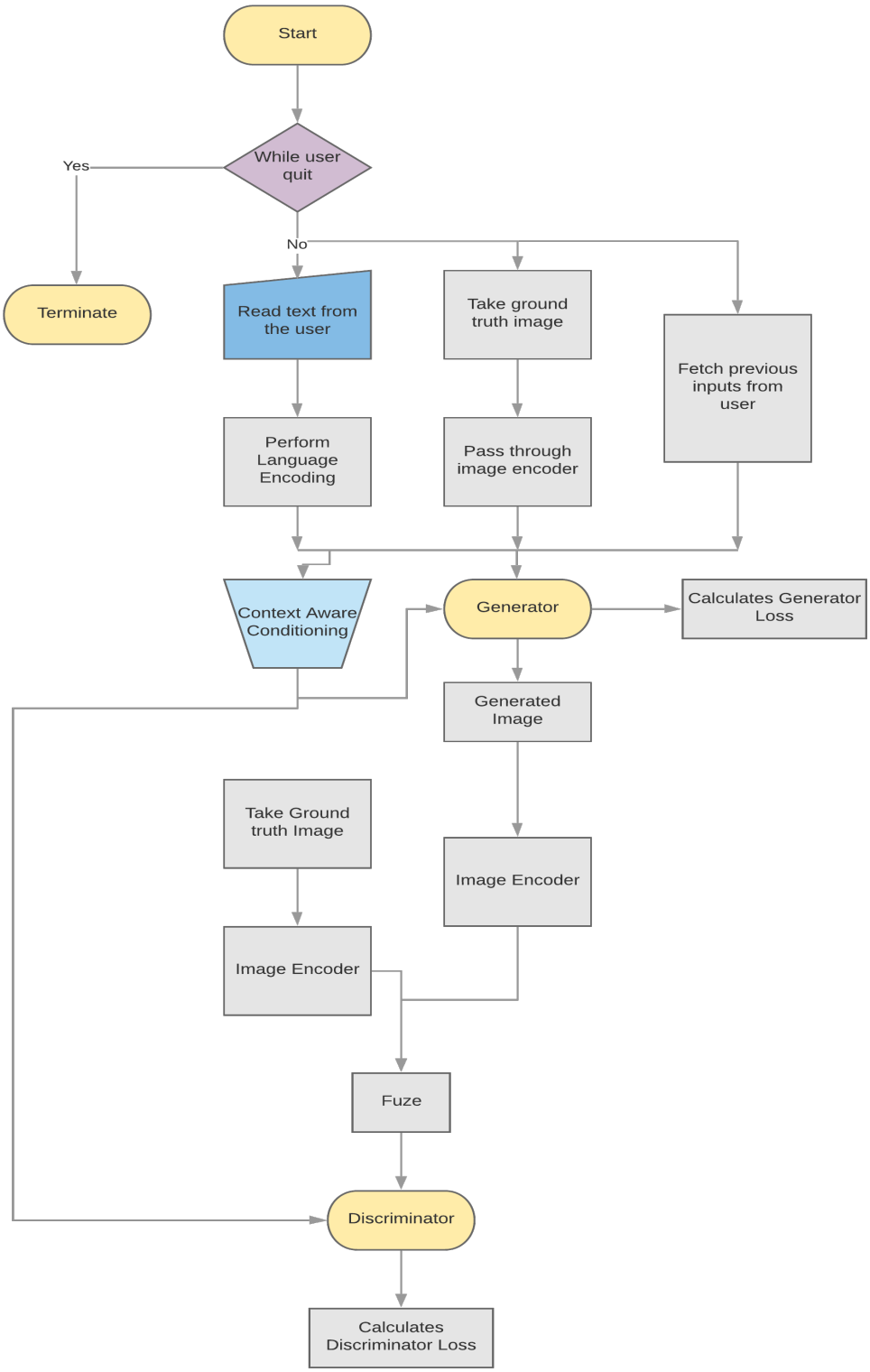

Figure 4.3: Data Flow for Training phase

Chapter 5

Testing and result analysis

We generated the dataset from the images and text captions in the form of Hierarchical Data Format (HDF) so that we can feed the data as a whole to the image encoder and thereafter to the model.

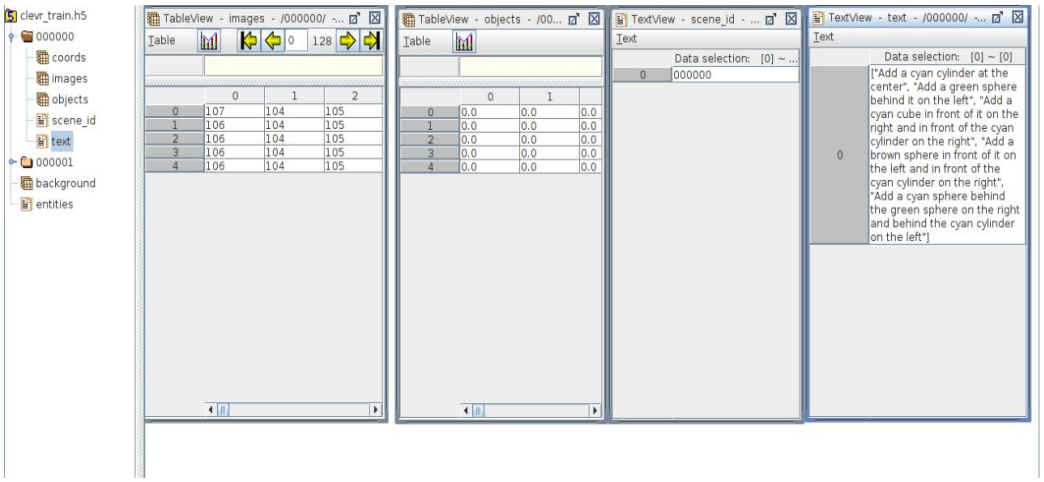

Figure 5.1: The HDF5 file structure after preparation of data.

```
Image Encoder model structure:

ImageEncoder(
  (image_encoder): Sequential(
    (0): ResDownBlock(
      (activation): ReLU()
      (downsampler): AvgPool2d(kernel_size=2, stride=2, padding=0)
      (conv1): Conv3x3(3, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv2): Conv3x3(64, 64, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv_shortcut): Conv1x1(3, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
    )
    (1): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (2): ResDownBlock(
      (activation): ReLU()
      (downsampler): AvgPool2d(kernel_size=2, stride=2, padding=0)
      (conv1): Conv3x3(64, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv2): Conv3x3(128, 128, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv_shortcut): Conv1x1(64, 128, kernel_size=(1, 1), stride=(1, 1), bias=False)
    )
    (3): BatchNorm2d(128, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (4): ResDownBlock(
      (activation): ReLU()
      (downsampler): AvgPool2d(kernel_size=2, stride=2, padding=0)
      (conv1): Conv3x3(128, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv2): Conv3x3(512, 512, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
      (conv_shortcut): Conv1x1(128, 512, kernel_size=(1, 1), stride=(1, 1), bias=False)
    )
    (5): BatchNorm2d(512, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  )
)
```

Figure 5.2: The Neural Structure of Image Encoder class.

The image data in the HDF file is fed to the image encoder which converts the image into feature vectors. The image encoder is used to encode the current time step image (real or generated) and the previous time-step ground-truth image. The output feature maps will be of the dimensions (Kd * Kd * Nd) which are passed through a fusion layer.

Chapter 6

Conclusion

We will use a recurrent GAN model for the task which is to be done and to show that the model is able to draw reasonable images for the provided instructions iteratively. The model should strictly outperform the normal non-iterative baseline model. So we would continue our work to implement the model to take iterative inputs from the user and generate the images iteratively like the scenes. Since this task can have several plausible solutions and no existing metric can capture all of them, we proposed a relational similarity metric to capture the possible relationships.

References

[1] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza,Bing Xu, David Warde-Farley, Sherjil Ozair, AaronCourville, and Yoshua Bengio, "Generative adversarial nets"

[2] Mehdi Mirza and Simon Osindero, "Conditional generative adversarial nets," `arXiv:1411.1784`

[3] Augustus Odena, Christopher Olah, and JonathonShlens,"Conditional image synthesis with auxiliary classifier GANs"

[4] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee, Generative adversarial text to image synthesis"

[5] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N. Metaxas, "StackGAN: Text to photo-realistic image synthesis with stacked generative adversarial networks"

[6] Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee, "Inferring semantic layout for hierarchical text-to-image synthesis"

[7] Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He, "AttnGAN: Fine-grained text to image generation with attentional generative adversarial networks"

[8] Shikhar Sharma, Dendi Suhubdy, Vincent Michalski,Samira Ebrahimi Kahou, and Yoshua Bengio, "Chatpainter: Improving text to image generation using dialogue"

[9] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh,Deshraj Yadav, Jos e M.F. Moura, Devi Parikh, andDhruv Batra, "Visual Dialog"

[10] Jianwei Yang, Anitha Kannan, Dhruv Batra, and DeviParikh, "LR-GAN: Layered recursive generative adver-sarial networks for image generation"

[11] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, EliShechtman, and Simon Lucey, “ST-GAN: Spatial trans-former generative adversarial networks for image com-positing”

Appendix A

Appendix

appendix here