EEE-443 Term Project
A Comparative Analysis of Text to Image Generation Models

Emirhan Tekez, Student ID: 22103385
Ahmet Arda Kocabağ, Student ID: 22003253
Berkay Altıntaş, Student ID: 22002709
Ozan Cem Baş, Student ID: 22102757
Mehmet Oral, Student ID: 22102156

January 10, 2025

Contents

List of Figures

List of Tables

Abstract

This study presents a comparative analysis of five text-to-image (T2I) generative models DF-GAN, AttnGAN, AttnGAN+CL, DM-GAN, and DM-GAN+CL which are pretrained on the MS COCO dataset. The primary objective is to determine the most suitable model under both quantitative and qualitative criteria. To achieve this, models are evaluated using the Fréchet Inception Distance (FID) score, VRAM usage during inference, and inference time per output image. Four sample prompts were applied to each model, and the resulting images were analyzed to obtain performance measures. Finally, the limitations and trade-offs of each model were discussed.

1 Introduction

The field of deep learning has witnessed significant advancements beyond classification tasks, particularly in the domain of generative models. Text-to-image synthesis, is one of the subfields of generative modelling that allows for the generation of visually coherent images from textual descriptions. This capability has been realized, initially, through the adoption and evolution of encoder-decoder architectures that were originally implemented in variational auto-encoder (VAE) structures [1]. The potential of text-to-image models lies in their conditional generative framework, where the generation process is guided not just by random noise but by structured inputs such as text. By encoding semantic information, models bridge the gap between textual and visual data, using techniques like RNNs or transformers for text encoding and CNNs or Vision Transformers (ViTs) for feature extraction [1]. This conditioning enables the synthesis of images that not only align with the description of the text but also allows capturing details. Advanced architectures integrate multiple input modalities—such as text, images, or their features—fusing these representations to produce outputs tailored to user specifications as illustrated in the figure 1.

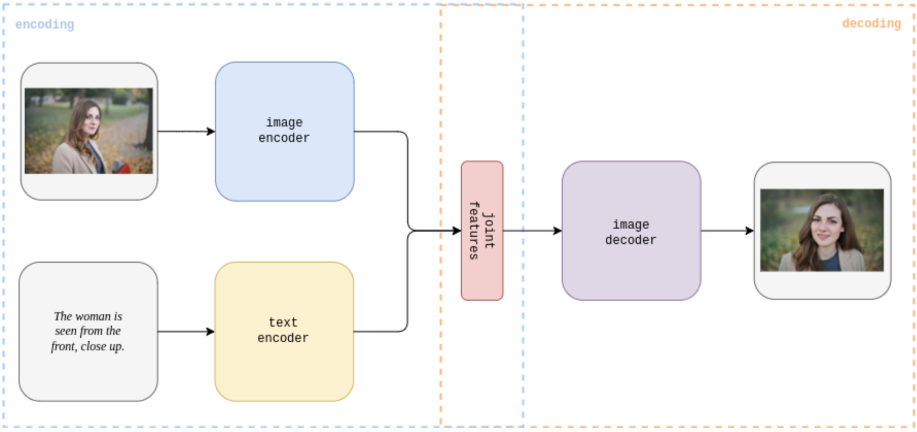


Figure 1: A basic illustration of text-to-image generation procedure [1].

Text-to-image generation models have witnessed rapid advancements over the past decade. Starting with alignDRAW in 2015, Generative Adversarial Networks (GANs) have been crucial in the further development of the field [2]. GANs were followed by the emergence of

transformer-based models such as Stable Diffusion [3], which are promising, yet are much slower in inference time and are heavier on the hardware requirements. A GAN consists of two neural networks: a generator and a discriminator, which operate in opposition to one another. The generator synthesizes data from random noise, aiming to produce outputs resembling real data. The output of this generator is then fed into the discriminator alongside real data samples, which tries to determine whether a given sample is real or generated. Both the discriminator and the generator are trained simultaneously, with the generator aiming to "deceive" the discriminator and the discriminator attempting to differentiate real from fake data. These two opposing optimization goals form the skeleton of all GAN models.

As mentioned above, GANs are faster in inference, as they do not have to process the data in multiple steps, such as denoising diffusion models which can take up to 1000 steps per sample. A single forward pass through the model makes the model faster as well as lighter in terms of computation. Storage-wise, GAN models are also smaller, as the models investigated in this study are $\sim$ 200MB, where as pretrained diffusion models can easily reach 5GB or more in size.

While lots of different GAN models are available online, this study focuses on the five following pretrained models: DF-GAN [4], AttnGAN [5], AttnGAN+CL [6], DM-GAN [7], and DM-GAN+CL [6]. The models were evaluated based on their Fréchet Inception Distance (FID) scores [8], VRAM usage during inference, and inference time per sample. At the end of the study, the tradeoffs in selecting any one of these models are discussed.

2 Methods

2.1 MS COCO Dataset

MS COCO 2017 (Microsoft Common Objects in Context) is a dataset containing RGB images, containing 123000 unlabeled images, and 164000 labeled images. The labeled images are further divided into 118000 training images, 5000 validation images, and 41000 testing images. The dataset is mostly used for object detection and classification, as well as image segmentation tasks. The annotations for the images are provided in the ".json" file format and contains information on which classes are located where on the image by providing the coordinates of bounding box vertices. The Python package "pycocotools" can be used to easily access the mentioned information from the given ".json" files. There's also information for segmentation purposes, where the dataset provides a pixelwise mask for image segmentation purposes as an array. The caption - image matches are not unique for this dataset, as there are five different captions per image [5].

2.2 Evaluation Environment

All five models were evaluated on Linux Ubuntu 20.04 Focal Fossa with CUDA 12.6. For Python packages, Pytorch 1.9.0, torchvision 0.10.0, scikit-image 0.18.0, and scipy 1.1.0 were used. The inference computer's hardware included a Nvidia GTX 1660Ti GPU with 6GB of VRAM, an AMD Ryzen 7 7700 CPU, and 32GB of system RAM.

2.3 Deep Fusion Generative Adversarial Network (DF-GAN)

DF-GAN is a simplified approach to text-to-image GANs, addressing key limitations in previous GAN-based methods. The model uses a bi-directional Long Short Term Memory (LSTM) network for text encoder part, alongside convolution layers for feature extraction from the generated or real images for image encoding purposes. The model improves upon existing solutions in the three following ways.

One-Stage Generator-Discriminator Pair: Unlike the stacked architecture used in existing methods such as StackGAN [9], DF-GAN employs a single-stage generator-discriminator pair. This eliminates complication between generators across scales and directly synthesizes high-resolution images with greater stability using hinge loss and residual networks. The single stage structure also reduces computational cost for both training and inference.

Target-Aware Discriminator: It introduces Matching-Aware Gradient Penalty (MA-GP) to regularize the discriminator, improving semantic consistency between text and images without the need for extra networks. Also, it replaces the traditional two-way output in discriminators with a One-Way Output, making the generator's convergence faster and ensuring better semantic alignment.

Deep Text-Image Fusion Block (DFBlock): It enhances text-image feature fusion through multiple stacked affine transformations blocks, combined with ReLU activations. The affine transformations include channel-wise scaling and shifting operations, deepening the fusion process, making it better for text descriptions while preventing computational inefficiencies due to cross-modal attention mechanisms. The architecture of DF-GAN is shown in figure 2.

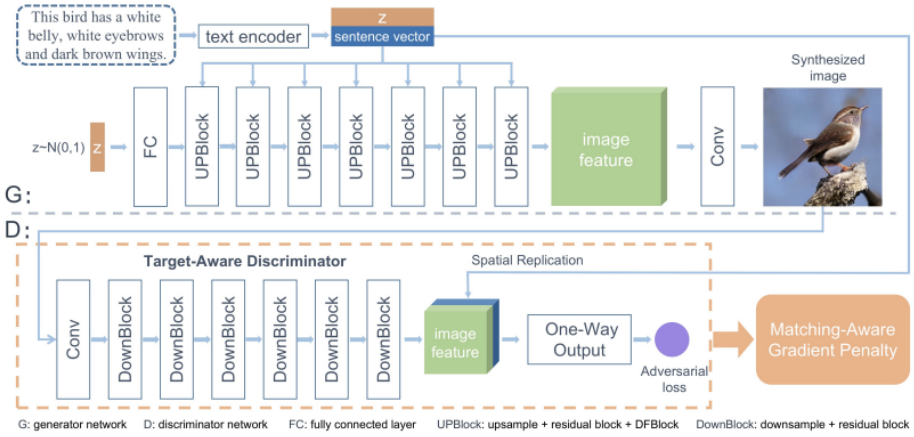


Figure 2: Architecture of DF-GAN [4]

2.4 Attentional Generative Adversarial Network (AttnGAN)

The AttnGAN introduces two key points distinguishing itself from the previous models:

Attentional Generative Network Unlike prior GAN-based methods that condition only on a single global sentence vector, AttnGAN employs word-level attention to focus on sub-regions of the image relevant to specific words in the description. It follows a multi-stage approach: a series of generators progressively clarifies the image. The first generator sketches relatively raw, low-resolution version conditioned on the global sentence vector; subsequent generators use attention over word embeddings to add fine details.

Deep Attentional Multimodal Similarity Model (DAMSM) DAMSM aligns text and image features in a shared semantic space at both global (sentence) and local (word) levels. It provides a fine-grained matching loss that ensures generated local regions correlate closely with the specific words describing them. More specifically, AttnGAN uses bi-directional LSTM in text encoder part to obtain semantic vectors from the texts and uses CNN in the image encoder part to map generated and real images to semantic vectors. This additional supervision (word-level alignment) stabilizes training and leads to more accurate, detailed image synthesis. Moreover, by adding the Contrastive Learning (CL) term to matching loss, one can use contrastive loss to maximizes the similarities between the matching text-image pair and minimizes the similarities between the not matching text-image pairs. In the figures 3, 4 and 5, architectures of DAMSM and DAMSM with CL and AttnGAN are provided respectively.

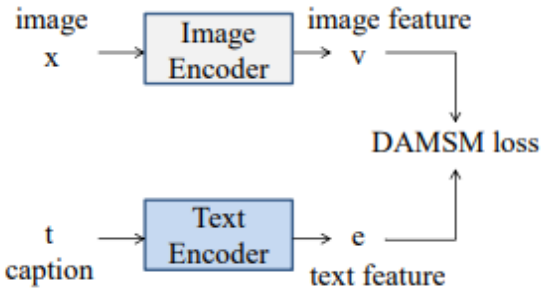


Figure 3: DAMSM Architecture [6]

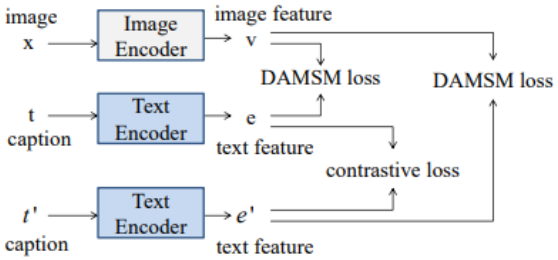


Figure 4: DAMSM CL Architecture [6]

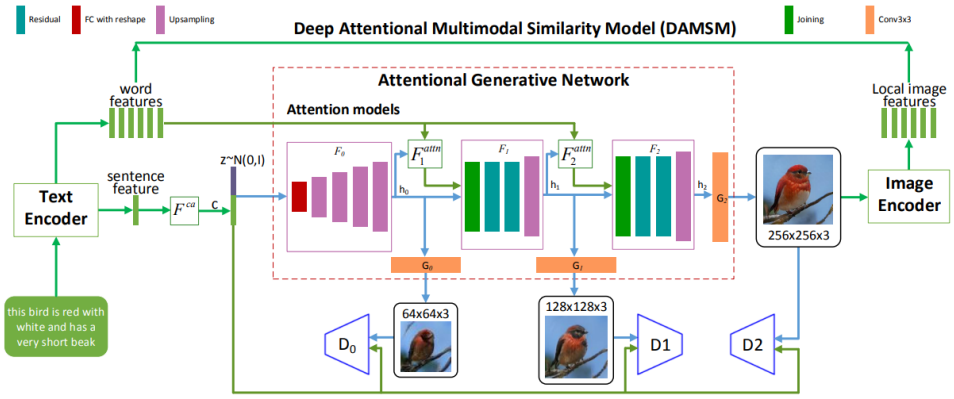


Figure 5: Architecture of AttnGAN [5]

2.5 AttnGAN with Contrastive Learning (AttnGAN+CL)

Contrastive Learning: It is an approach that learns representations of data by fetching similar data points closer in the embedding space while pushing dissimilar ones farther apart. It relies on contrasting positive pairs, derived from the same data point, and negative pairs, from unrelated data. This method is widely used in self-supervised learning to infer meaningful features without labeled data. Positive pairs can be obtained by applying random transformations such as resizing, cropping and rotation. By optimizing contrastive loss, models achieve robust and generalizable representations, enabling effective learning from large, unlabeled datasets.

Benefits of CL in AttnGAN:

Enhanced Semantic Consistency By aligning textual and visual features during pre-training and GAN training, contrastive learning ensures the synthetic images better reflect the fine details in input captions.

Improved Performance Metrics Experimental results demonstrate that adding contrastive learning improves metrics like FID, particularly on complex datasets like MS-COCO, where the captions are quite complex, including several adjectives and longer strings $\sim$ 20 words.

Improved Generalization The embedded representations from the contrastive pre-training stage ensure better generalization of text-to-image synthesis for unseen captions, and different datasets.

2.6 Dynamic Memory Adversarial Network (DM-GAN)

Previous models face two main issues: they rely too heavily on an initial low-resolution image, and they assume all words in a sentence are equally important—keeping the text

representation the same throughout. DM-GAN solves these problems by introducing three key innovations:

Dynamic Memory Module: A key-value memory structure stores text features and selectively reads them to correct and enrich fuzzy image regions. This allows the network to inject the most relevant word-level information into the image refinement process.

Memory Writing Gate: Adjusts how text features (words) are stored in the memory, gating the importance of each word based on the current image content.

Response Gate: The response gate controls how features from the dynamic memory module are fused with the evolving image features, which helps the model adaptively combine image and text cues. In figure 6, architecture of DM-GAN is provided.

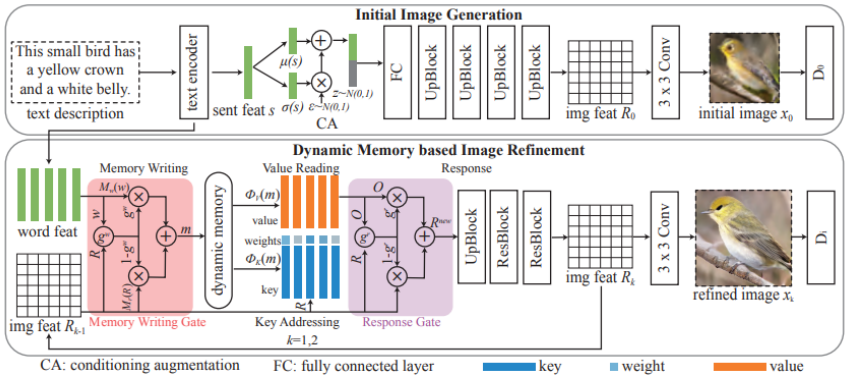


Figure 6: Architecture of DM-GAN [7]

2.7 DM-GAN with Contrastive Learning (DM-GAN+CL)

DM-GAN relies on an initial image feature for iterative refinement, but it does not fully address the scenario where multiple captions describe the same image with subtle linguistic variations. By introducing a contrastive learning framework, it was encouraged consistency across all captions linked to the same ground-truth image.

2.8 Fréchet Inception Distance (FID)

A quantitative way of judging the performance of generative models is Fréchet Inception Distance or FID. This metric takes in a large number of samples from both the generated images and the real data, and measures the difference in both the mean, and the variances of the produced outputs. A smaller score (or distance) indicates a better performing model which yields more realistic outputs. For this work, $\sim$ 20000 images were generated from each model, and these generated images' distribution were compared against the MS-COCO validation set to calculate the final FID score. The number of generated images was kept high in order to obtain a score which is more representative of the model's performance.

3 Results

3.1 Quantitative Results

Table 1: Quantitative Results of the GAN Models

Method	Dataset	FID Score↓	VRAM Usage↓	Sec/Inference Sample ↓
DF-GAN	MS-COCO	**15.47**	5623MB	**0.02378s**
AttnGAN	MS-COCO	33.98	**3994MB**	0.05460s
AttnGAN + CL	MS-COCO	24.29	4484MB	0.03432s
DM-GAN	MS-COCO	26.58	4079MB	0.04023s
DM-GAN + CL	MS-COCO	20.56	4176MB	0.04299s

Table 1 illustrates the quantitative results of the GAN models. The performance of the five COCO-only pre-trained T2I models were evaluated using the FID score metric, VRAM usage during the inference of the $\sim$ 20000 images mentioned above, and inference time per sample over the COCO dataset. For all evaluation metrics, "↓" is used if a smaller value indicates a better performance. Furthermore, the best achieved scores are indicated using the bold fonts.

According to Table 1, DF-GAN achieves the best performance in terms of FID score and inference time, which is 15.47 and 0.02378s respectively, compared to other pre-trained models. However, it gets also the highest VRAM usage, which is 5623 MB since it tries to generate high quality images with its deep text-image fusion blocks. DF-GAN's significantly reduced inference times also align with the expectations set above, as it only has one generator-discriminator pair, as opposed having a stacked structure with multiple layers.

According to the Table 1, it could be observed that the models with Contrastive Learning performs better than models without Contrastive Learning. To be more precise, it could be observed that although the FID score of the AttnGAN is 33.98, the FID score of the AttnGAN + CL is 24.29 which means the AttnGAN model with CL outperforms the AttnGAN. Similarly, even though the FID score of the DM-GAN is 26.58, the FID score decreases to 20.56 when CL is added, which means DM-GAN model with CL outperforms the standard DM-GAN. However, this is not the case for VRAM usage. Since the adding CL component to the standard architecture makes the architecture more complex, the VRAM usage of the models with CL components is greater than the VRAM usage of the standard models, which could lead to inference difficulties in situtations where the computation power is limited. Thus, it could be concluded that there is an inverse relationship between VRAM usage and FID score which represent the performance of the model.

3.2 Qualitative Results

Each model was tested with the following four previously unseen prompts below. Each model generates an images with a resolution of 256x256. Moreover, the corresponding generated images for the given prompts are compared in Table 2.

Prompt 1: A green car and a red motorcycle racing down the highway.

Prompt 2: A kid picking yellow flowers in the garden.

Prompt 3: A blue and red plane taking off from an airport at sunrise.

Prompt 4: A group of people playing football in a school garden.

Table 2: Comparison of GAN Output Images for Different Prompts

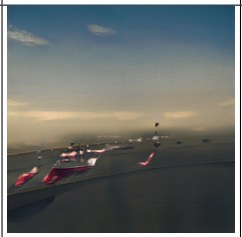
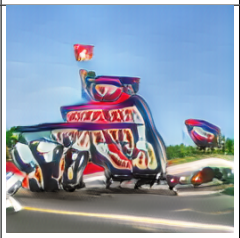
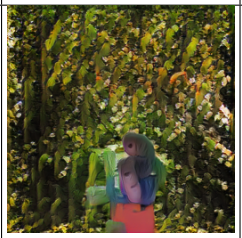
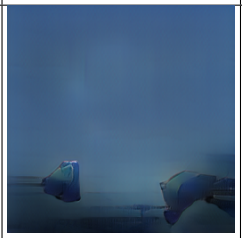
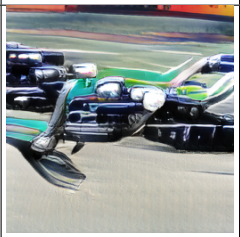
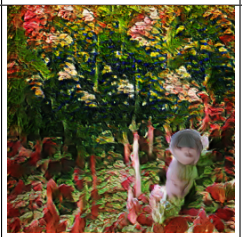
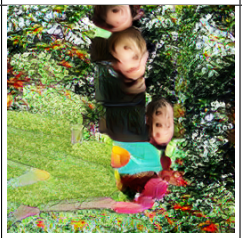
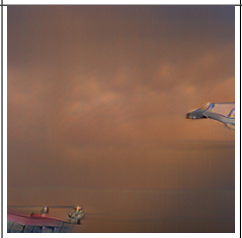
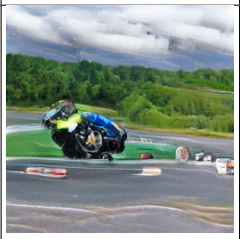
Model	Prompt 1	Prompt 2	Prompt 3	Prompt 4
DF-GAN				
AttnGAN				
AttnGAN+CL				
DM-GAN				
DM-GAN+CL				

Table 2 illustrates previously unseen prompts. In the first prompt, the focus was on generating distinct objects with varying colors to evaluate each model's ability to accurately

depict diverse object-color combinations. The second prompt aimed to create a scenario involving a person interacting with an outdoor environment, allowing the assessment of how well the models capture living being-environment interactions. For the third prompt, the main objective was to test the model's performance on large-scale scenes and objects. Lastly, the fourth prompt evaluated the models' performance in depicting interactions between multiple living beings, in order to visualize how well the models learned human-human interactions, and placing individuals in distinct positions within an image.

4 Discussion

In this study, five different COCO-only pre-trained T2I models, which are DF-GAN, AttnGAN, AttnGAN + CL, DM-GAN and DM-GAN + CL, were investigated. Since pre-trained models were used, the obtained results were similar to the results stated in the official implementations. Each model balances image quality (FID), computational efficiency (Seconds per Inference Sample), and memory (VRAM) usage differently . DF-GAN is ideal for high-quality, fast generation, while CL supported models show better consistency at increased resource costs. Future work could focus on optimizing resource usage, hybrid architectures, and testing on diverse datasets.

Qualitative analysis backs up the quantitative results, as DF-GAN, which had the highest FID score among the models, is the only model with a visually distinguishable car in the first prompt's picture. The models which used CL seem to be better at drawing large scenery, as they were the only 2 models which gave images of airports on the third prompt. The best performer on the final prompt is once again DF-GAN, which produced the most realistic human beings. The model was also able to place the individuals some distance apart from one another, whereas other models produce large "blobs" of humans in one area. Although there is some resemblance to real images, even the "best" model in terms of FID score, which is DF-GAN, struggles with drawing human beings. Body parts overlap with one another. The other models perform even worse, as it could be seen in the outputs of the Prompt 2 and Prompt 4. The generated images cannot distinguish the 'kids' and 'humans'. Moreover, it is hard to detect the number of people when the prompt says 'a group of people'. Regarding these, the further works may focus on the generating multiple objects or human beings clearly rather than creating realistic environments.

In general, it could be observed that each pre-trained model performs well to generate specific environments and weather conditions which were generally described in the prompts. Moreover, each model reflects the colours specified in the prompt succesfully. However, these models also suffer from their inability to generate multiple objects at the same time.

From these observations, it could be said that adding CL to GAN models tends to improve their performance. It could also be argued that adding CL to the DF-GAN model, which has the best FID score, could further improve its performance, though there would be a tradeoff, as the model's VRAM usage, and computational complexity would increase.

In order to improve the text embedding, and as a result, the semantic relation between the input caption and the output image, models such as CLIP and GloVe which are pre-trained on large datasets could be used, at the cost of increased computational complexity [10].

References

[1] Maciej Żelaszczyk and Jacek Mańdziuk. *Text-to-Image Cross-Modal Generation: A Systematic Review*. 2024. arXiv: 2401.11631 [cs.CV]. URL: https://arxiv.org/abs/2401.11631.

[2] Elman Mansimov et al. *Generating Images from Captions with Attention*. 2016. arXiv: 1511.02793 [cs.LG]. URL: https://arxiv.org/abs/1511.02793.

[3] M. Chew and M. Hakam. "Review of the Stable Diffusion Model: Generative Image Synthesis Using Latent Diffusion Models". In: *Open Science Framework (OSF)* (Dec. 2024). DOI: 10.31219/osf.io/gdb2f. URL: https://doi.org/10.31219/osf.io/gdb2f.

[4] Ming Tao et al. *DF-GAN: A Simple and Effective Baseline for Text-to-Image Synthesis*. 2022. arXiv: 2008.05865 [cs.CV]. URL: https://arxiv.org/abs/2008.05865.

[5] Tao Xu et al. *AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks*. 2017. arXiv: 1711.10485 [cs.CV]. URL: https://arxiv.org/abs/1711.10485.

[6] Hui Ye et al. *Improving Text-to-Image Synthesis Using Contrastive Learning*. 2021. arXiv: 2107.02423 [cs.LG]. URL: https://arxiv.org/abs/2107.02423.

[7] Minfeng Zhu et al. *DM-GAN: Dynamic Memory Generative Adversarial Networks for Text-to-Image Synthesis*. 2019. arXiv: 1904.01310 [cs.CV]. URL: https://arxiv.org/abs/1904.01310.

[8] D. C. Dowson and B. V. Landau. "The Fréchet distance between multivariate normal distributions". In: *Journal of Multivariate Analysis* 12.3 (1982), pp. 450–455. DOI: 10.1016/0047-259x(82)90077-x. URL: https://doi.org/10.1016/0047-259x(82)90077-x.

[9] Han Zhang et al. *StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks*. 2017. arXiv: 1612.03242 [cs.CV]. URL: https://arxiv.org/abs/1612.03242.

[10] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV]. URL: https://arxiv.org/abs/2103.00020.

5 Appendix

5.1 Figures for Quantitative Results

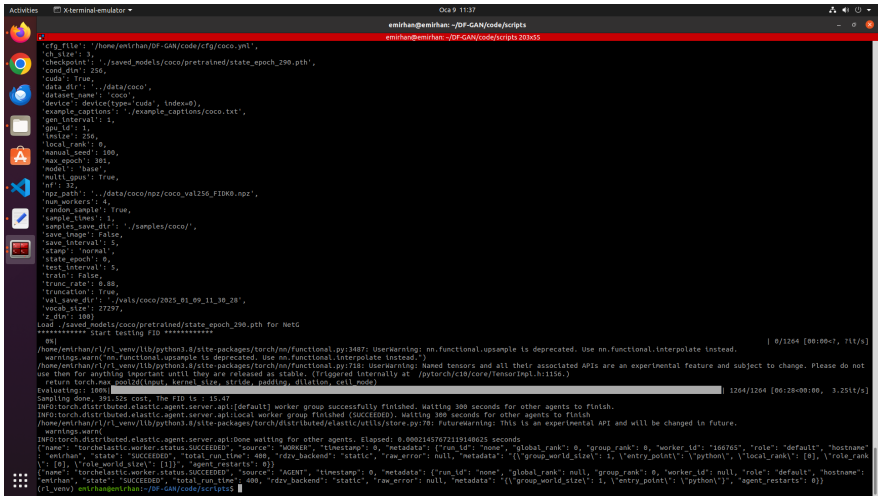


Figure 7: FID Score of DF-GAN and Inference Duration

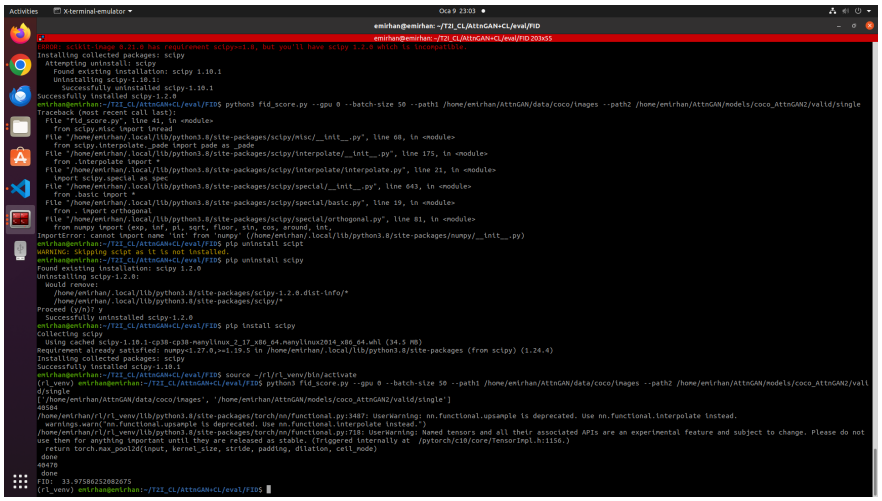


Figure 8: FID Score of AttnGAN

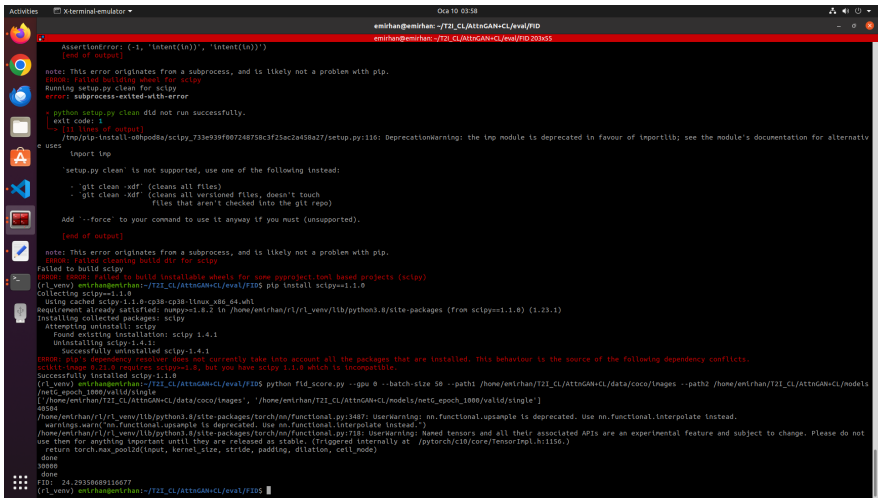


Figure 9: FID Score of AttnGAN+CL

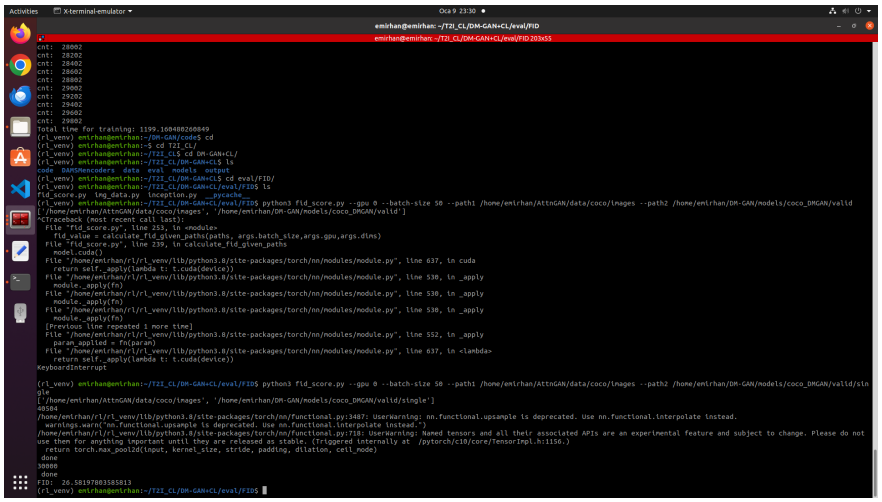


Figure 10: FID Score of DM-GAN

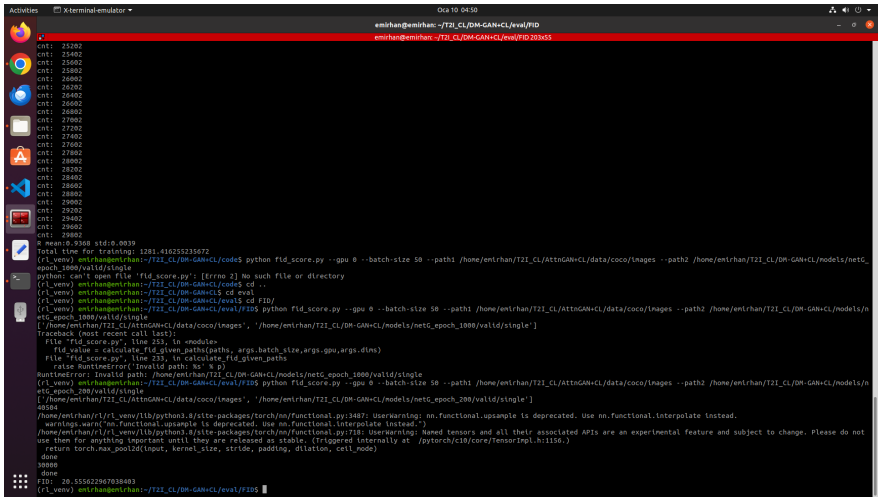


Figure 11: FID Score of DM-GAN+CL

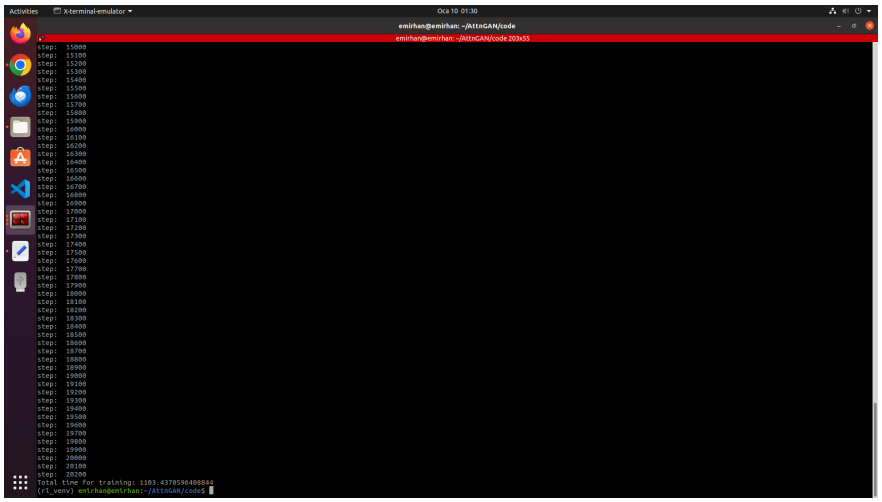


Figure 12: Inference Time of AttnGAN for 29802 Images

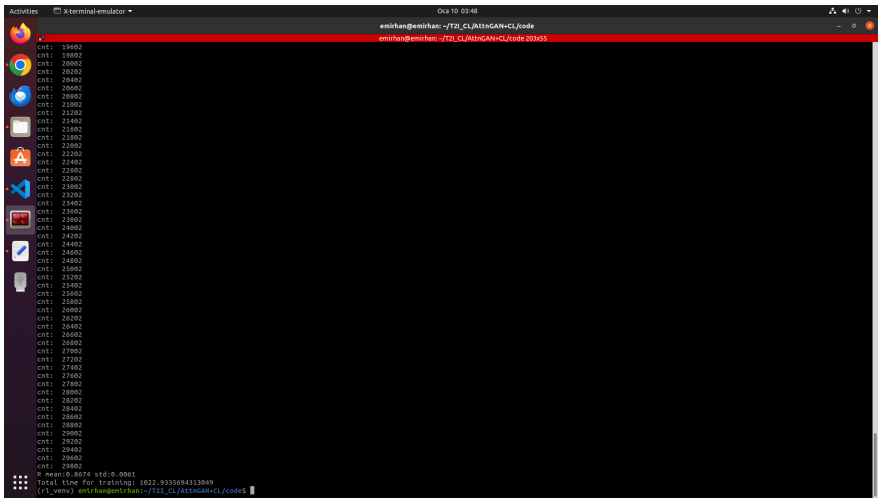


Figure 13: Inference Time of AttnGAN+CL for 29802 Images

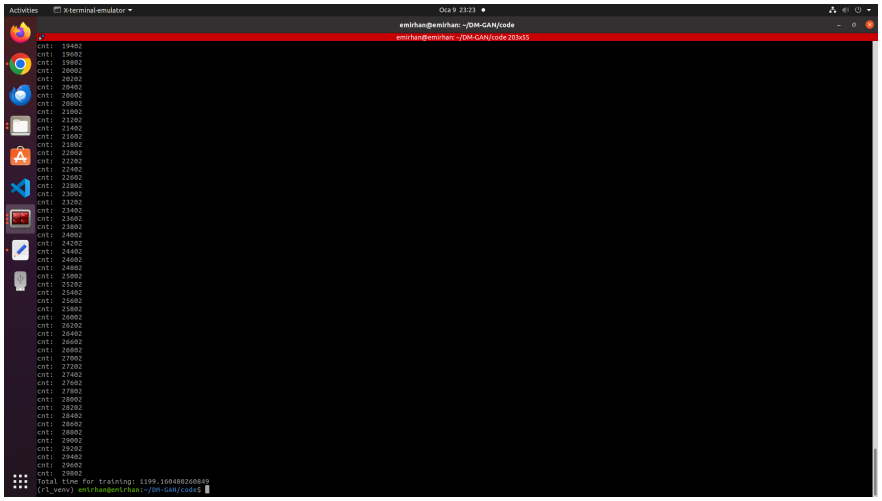


Figure 14: Inference Time of DM-GAN for 29802 Images

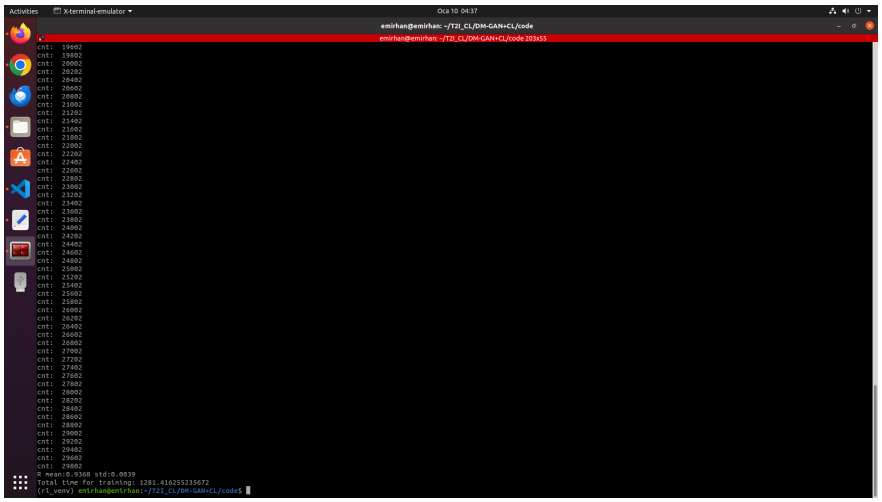


Figure 15: Inference Time of DM-GAN+CL for 29802 Images

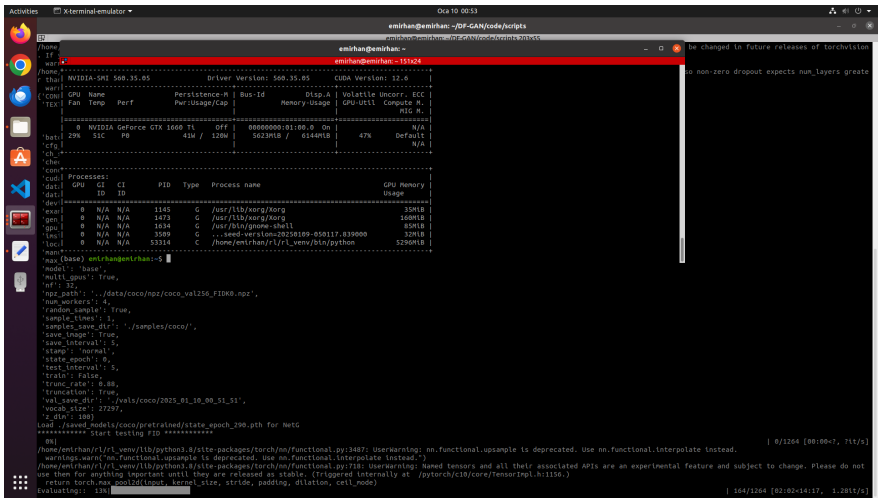


Figure 16: VRAM Consumption of DF-GAN for Inferring 20002 Images

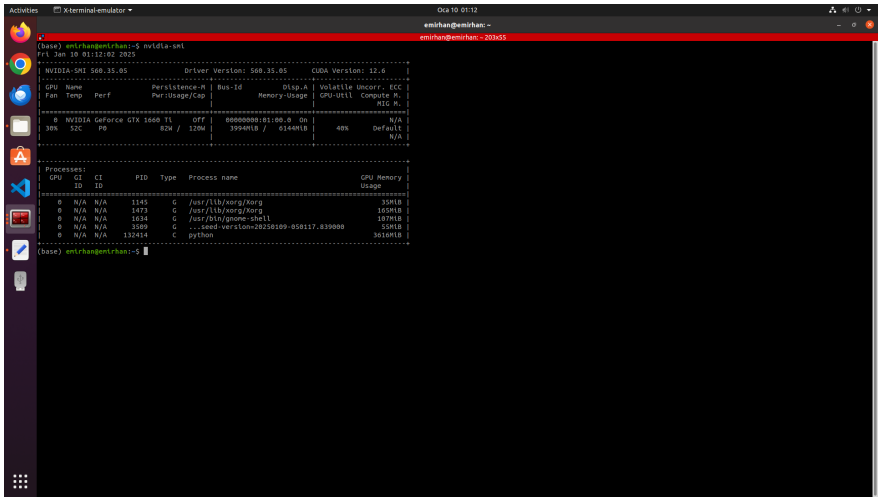


Figure 17: VRAM Consumption of AttnGAN for Inferring 29802 Images

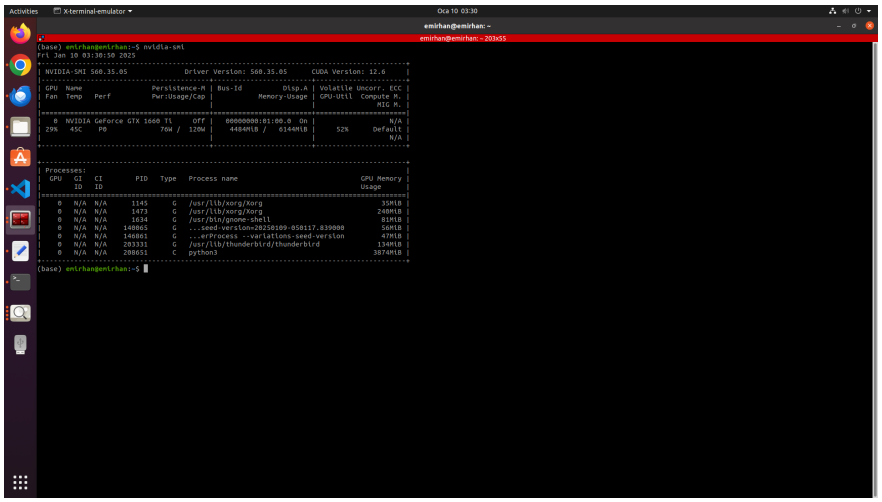


Figure 18: VRAM Consumption of AttnGAN+CL for Inferring 29802 Images

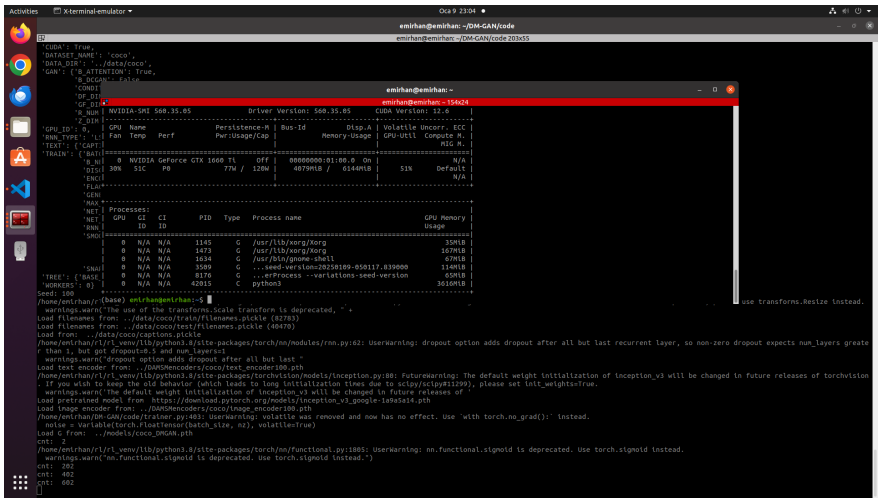


Figure 19: VRAM Consumption of DM-GAN for Inferring 29802 Images

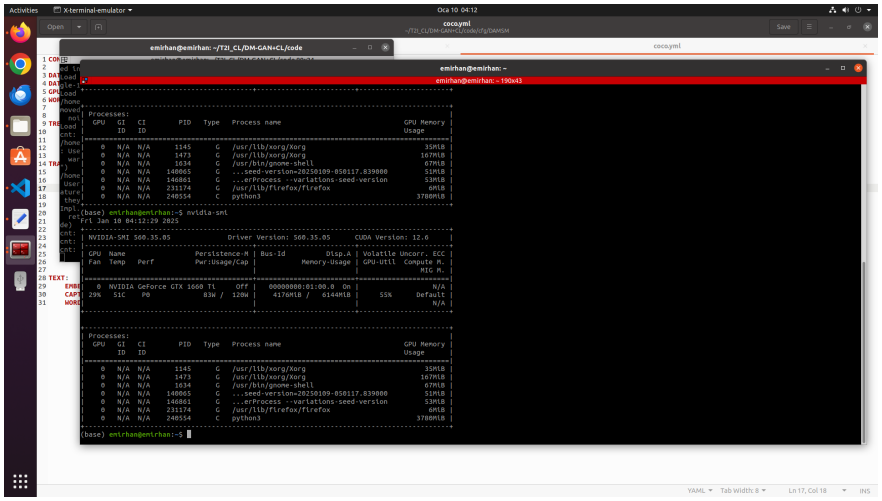


Figure 20: VRAM Consumption of DM-GAN+CL for Inferring 29802 Images

5.2 Links for Evaluated Models

For the pre-trained model parameters:

1. DAMSM: https://drive.google.com/file/d/1zIrXCE9F6yfbEJIbNP5-YrEe2pZcPSGJ/view
2. DMGAN: https://drive.google.com/file/d/1tQ9CJNiLlRLBKSUKHXKYms2tbfzllyO-/edit
3. AttnGAN + CL: https://drive.google.com/file/d/1nNB-MHGkVLWj1zlOcsDVGzyhkrsvw7UY/view
4. DMGAN + CL: https://drive.google.com/file/d/1nNB-MHGkVLWj1zlOcsDVGzyhkrsvw7UY/view
5. AttnGAN: https://drive.google.com/file/d/1i9Xkg9nU74RAvkcqKE-rJYhjvzKAMnCi/view

The Github links to the models:

1. AttnGAN and DAMSM: https://github.com/davidstap/AttnGAN
2. DFGAN: https://github.com/tobran/DF-GAN?tab=readme-ov-file
3. DMGAN: https://github.com/JuhongPark/DM-GAN
4. AttnGAN + CL and DMGAN + CL: https://github.com/huiyegit/T2I_CL/tree/main

5.3 Codes for The Project

```
git clone https://github.com/tobran/DF-GAN.git
pip install -r requirements.txt
cd DF-GAN/code/
bash scripts/calc_fid.sh ./cfg/coco.yml
bash scripts/sample.sh ./cfg/coco.yml

cd ~
git clone https://github.com/davidstap/AttnGAN.git
cd AttnGAN/code/
python main.py --cfg cfg/eval_coco.yml --gpu 0
cd ..
cd eval
cd FID
python fid_score.py --gpu 0 --batch-size 50 --path1 /home/emirhan/
   T2I_CL/AttnGAN/data/coco/images --path2 /home/emirhan/AttnGAN/
   models/coco_AttnGAN2/single


cd~
git clone https://github.com/JuhongPark/DM-GAN.git
cd DM-GAN/code
python main.py --cfg cfg/eval_coco.yml --gpu 0
cd ..
cd eval
cd FID
python fid_score.py --gpu 0 --batch-size 50 --path1 /home/emirhan/
   DM-GAN/data/coco/images --path2 /home/emirhan/DM-GAN/models/
   coco_DMGAN/valid/single




cd~
git clone https://github.com/huiyegit/T2I_CL.git
cd T2I_CL/AttnGAN+CL/code
python main.py --cfg cfg/eval_coco.yml --gpu 0
cd ..
cd eval
cd FID
python fid_score.py --gpu 0 --batch-size 50 --path1 /home/emirhan/
   T2I_CL/AttnGAN+CL/data/coco/images --path2 /home/emirhan/T2I_CL
   /AttnGAN+CL/models/netG_epoch_1000/valid/single


cd~
cd T2I_CL/DM-GAN+CL/code
```

```
python main.py --cfg cfg/eval_coco.yml --gpu 0
cd ..
cd eval
cd FID
python fid_score.py --gpu 0 --batch-size 50 --path1 /home/emirhan/
   T2I_CL/DM-GAN+CL/data/coco/images --path2 /home/emirhan/T2I_CL/
   DM-GAN+CL/models/netG_epoch_200/valid/single
```