

CLIP-Guided Zero-Shot Text-to-Image Generation

Project Description

In this project, you will assign generative capabilities to a Generative Adversarial Network (GAN) by employing CLIP (Contrastive Language–Image Pretraining) as a scoring function for image generation. The GAN framework consists of a generator and a discriminator trained in a minimax setting, where the generator learns to map latent codes z sampled from a prior distribution to realistic images. We employ StyleGAN2-ADA [2], a style-based variant that introduces adaptive discriminator augmentation to stabilize training and improve image fidelity, especially under limited data conditions.

CLIP [3], on the other hand, is a multimodal model trained to align images and text in a joint embedding space through contrastive learning over large-scale image–caption pairs. It provides a semantic consistency measure between text and image features via cosine similarity. In this setup, CLIP serves as an external objective guiding the generator toward producing images that are semantically aligned with a given text prompt. This enables CLIP-guided latent optimization for semantically aligned face generation, where the text prompt steers the generator toward visually coherent attributes (e.g., gender, age, or expression) without retraining.

You are provided with a pretrained StyleGAN2-ADA generator and the pretrained CLIP model to use throughout the project. A sample Jupyter notebook is also included to illustrate how to synthesize images from latent codes using the pretrained StyleGAN2-ADA model and how to compute CLIP similarity scores between images and text prompts. Your task is to extend these examples to implement CLIP-guided image generation, bias and

fairness analysis, and latent space exploration as described in the following parts.

Note: This project does not require a training process. However, it is recommended to use **Kaggle** for its computational resources.

Note: For this project, a pre-trained **StyleGAN2-ADA** (FFHQ) model is provided for inference. Students are not required to train any model. The model can be downloaded from the following link:

https://drive.google.com/file/d/1vYu8UxAMtQZr08wCImpf1-yxBgE-1V7U/view?usp=drive_link

For any questions regarding this project, you may e-mail ergunesr@itu.edu.tr.

Part I: CLIP-Guided Image Generation (40 Pts)

In this part, you are expected to implement the CLIP-guided image generation process. The goal is to sample diverse latent vectors, generate corresponding images using a pretrained GAN, and evaluate their semantic alignment with a given text prompt using CLIP similarity scores.

1. Generate a set of images from different random latent vectors (use a conditional GAN or any GAN that accepts a latent vector z and optional condition vectors).
2. Encode each generated image using CLIP.
3. Encode the text prompt with CLIP.
4. Compute the similarity between each image embedding and the text embedding.
5. Select the image with the highest similarity as the final output.

Discuss: How does the best similarity score change as the number of generated samples increases?

Part II: Bias and Fairness Analysis (30 Pts)

In this part, you will analyze potential bias and fairness issues in the employed GAN model. By using different attribute categories and prompts, you are expected to cluster CLIP image embeddings for these pairs and discuss any visible grouping or bias patterns.

For example, by examining occupation-related prompts, the model’s bias toward gender or age can be identified. Question: Are the images generated for “CEO” prompts more similar to “male” embeddings?

1. Generate multiple samples per prompt.
2. Encode generated images using CLIP.
3. Encode text prompts such as “a man” and “a woman” as reference points.
4. Measure cosine similarities to determine whether one cluster is biased toward a particular gender.

Investigate:

- **Representation consistency:** Are the embeddings for “female CEO” visually coherent, or does the model occasionally fail (e.g., produce non-female faces)?
- **Latent separation:** Do “male CEO” and “female CEO” embeddings form distinct clusters even though the profession is the same?

Bias Score Definition

The following equation defines the *Bias Score*, which compares cosine similarities between generated image embeddings and gender-related text embeddings.

A positive Bias Score indicates that the generated images are, on average, more semantically aligned with male-related text embeddings, whereas a negative score reflects stronger alignment with female-related embeddings. Values near zero suggest balanced representations, implying that the generative model does not exhibit a strong gender bias in CLIP’s embedding space:

$$\text{BiasScore} = \frac{1}{N} \sum_{i=1}^N [\cos(v_i, t_{\text{man}}) - \cos(v_i, t_{\text{woman}})] \quad (1)$$

Discuss your observations and report the computed Bias Scores for each case.

Attribute Categories and Prompts

Tables 1 and 2 provide example prompt categories for generating diverse and interpretable image samples. Table 1 focuses on visual and perceptual attributes such as facial features, lighting, and pose, whereas Table 2 lists prompts capturing social and demographic aspects used for fairness analysis.

Category	Example Prompts
Facial Attributes	“A person wearing glasses”, “A person not wearing glasses”, “A person with curly hair”, “A person with straight hair”, “A bald person”, “A person with short hair”, “A person with long hair”.
Expressions	“A smiling person”, “A person looking serious”, “A person laughing”, “A surprised person”, “A sad person”.
Accessories & Clothing	“A person wearing a hat”, “A person wearing earrings”, “A person wearing a suit”, “A person wearing a hoodie”, “A person wearing a scarf”.
Age Groups	“A young person”, “An old person”, “A baby”, “A middle-aged person”.
Pose / Angle	“A profile photo of a person”, “A portrait photo of a person facing the camera”, “A person looking to the side”.
Lighting / Color	“A black and white portrait of a person”, “A person under dramatic lighting”, “A person in a dark environment”.

Table 1: Visual attribute categories used for prompt generation.

Category	Example Prompts
Occupations	“A photo of a CEO”, “A photo of a nurse”, “A photo of a scientist”, “A photo of a teacher”, “A photo of a police officer”, “A photo of a chef”, “A photo of a farmer”, “A photo of a soldier”.
Gender Variations	“A female CEO”, “A male CEO”, “A female scientist”, “A male scientist”, “A female teacher”, “A male teacher”.
Ethnicity Variations	“A portrait of an Asian person”, “A portrait of a Black person”, “A portrait of a White person”, “A portrait of a Middle Eastern person” (used only for educational analysis and visualization with ethical framing).
Age/Gender Mix	“An old man”, “An old woman”, “A young man”, “A young woman”.
Emotional Attributes	“A confident person”, “A shy person”, “A tired person”.
Fairness Comparisons	“A person in a leadership role” vs. “A person in a supportive role”, “A person with glasses” vs. “A person without glasses”.

Table 2: Social and demographic attribute categories used for bias and fairness evaluation.

Part III: Latent Space Analysis (30 Pts)

In this part, you are expected to explore whether semantic directions in the latent space correspond to interpretable or biased transformations. By analyzing differences in latent codes associated with specific attributes, you will examine how controllable factors (e.g., presence of glasses, gender, or age) emerge in the learned representation space of StyleGAN2-ADA.

StyleGAN models map the input latent vector z (sampled from a standard normal distribution) to an intermediate latent space \mathcal{W} through a mapping network f . This intermediate space is designed to improve disentanglement, meaning that single directions in \mathcal{W} often correspond to meaningful visual changes in the generated image (e.g., hairstyle, expression, or gender).

1. Sample latent codes $z_i \sim \mathcal{N}(0, I)$ and transform them into the \mathcal{W} space using the mapping network $w_i = f(z_i)$. Generate corresponding images $x_i = G(w_i)$ using the pretrained generator G .

2. For each generated image, compute its CLIP embedding v_i and evaluate cosine similarities with the target text embeddings, e.g., $\cos(v_i, t_{\text{glasses}})$ and $\cos(v_i, t_{\text{no-glasses}})$.
3. Group the latent codes based on the most semantically aligned prompt and compute the average latent representation for each attribute:

$$\bar{w}_{\text{glasses}} = \frac{1}{K} \sum_{i=1}^K w_i^{(\text{glasses})}, \quad \bar{w}_{\text{no-glasses}} = \frac{1}{K} \sum_{i=1}^K w_i^{(\text{no-glasses})}.$$

4. Estimate the semantic direction in latent space as the difference between mean representations:

$$d_{\text{glasses}} = \bar{w}_{\text{glasses}} - \bar{w}_{\text{no-glasses}}.$$

This vector d_{glasses} captures the direction in \mathcal{W} that corresponds to adding or removing glasses in the generated output.

5. To visualize the effect of this direction, perform a latent traversal starting from a base latent code w_{base} :

$$w' = w_{\text{base}} + \alpha d_{\text{glasses}},$$

where α controls the intensity of the manipulation (positive α increases the presence of glasses, negative α removes them). Generate images $G(w')$ for varying α to visualize the continuous transformation.

This process helps reveal how the GAN encodes semantic attributes in its latent space and whether these directions are disentangled (affecting one attribute at a time) or entangled (affecting multiple correlated features such as gender or age).

Bonus Section: Perceptual Path Length (PPL) (Bonus 20 Pts)

To further assess latent space smoothness, compute or visualize the Perceptual Path Length (PPL) [1]. PPL measures the perceptual change in generated images along interpolated latent paths. Lower PPL values indicate smoother transitions and more disentangled latent factors. Discuss how interpolation smoothness reflects the structure of the learned latent manifold.

Appendix A: Environment Setup and Requirements

This project relies on **StyleGAN2-ADA** (Karras et al., *NeurIPS 2020*), an improved variant of StyleGAN2 designed to maintain high-fidelity image synthesis under limited data regimes. StyleGAN2-ADA introduces **Adaptive Discriminator Augmentation (ADA)**, a dynamic regularization mechanism that prevents discriminator overfitting, resulting in a more stable and data-efficient training process.

In this project, a pretrained StyleGAN2-ADA generator is provided, together with a **CLIP** model for semantic alignment between text and image embeddings. The accompanying Jupyter notebook illustrates how to load the pretrained generator, synthesize images from random latent codes, and compute CLIP-based similarity scores for zero-shot text-to-image evaluation.

Required Dependencies

The following packages are required to execute the notebook successfully:

```
torch>=1.9.0
torchvision>=0.10.0
numpy
Pillow
scipy
tqdm
requests
click
matplotlib
imageio
ftfy
regex
git+https://github.com/openai/CLIP.git
```

To install all dependencies, use either:

```
pip install -r requirements.txt
```

or manually:

```
pip install torch torchvision numpy Pillow scipy tqdm requests click \
matplotlib imageio ftfy regex git+https://github.com/openai/CLIP.git
```

Appendix B: Deliverables

By the end of this project, you are expected to submit a well-documented report and accompanying code that demonstrate your implementation, analysis, and findings. Your submission should include the following components:

1. **Project Report (PDF)** The report should include:
 - A brief description of your implementation for CLIP-guided image generation (Part I).
 - The set of prompts used, representative generated images, and corresponding CLIP similarity scores.
 - A discussion of your findings regarding bias and fairness analysis (Part II), supported by visualizations or clustering results.
 - Reported Bias Scores for different categories and a discussion of their implications.
 - Latent space traversal results (Part III), including sample images illustrating semantic directions and smoothness evaluation (optional PPL analysis).
 - A short conclusion summarizing key observations and limitations.
2. **Code and Notebook Files** The complete source code used for your experiments must be submitted as:
 - A Jupyter Notebook (.ipynb) demonstrating your main implementation steps.
 - Any helper scripts or modules (.py files) required to reproduce your results.

All files should be organized under a single directory and compressed into a .zip file named as: <StudentID>_Project.zip

Example: 150123456_Project.zip

References

- [1] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

- [2] Tero Karras, Miika Aittala, Janne Hellsten, Samuli Laine, Jaakko Lehtinen, and Timo Aila. Training generative adversarial networks with limited data. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.
- [3] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamila Mishkin, et al. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021.