# Generating Images Using Textual Inversion

**By**

Baker Iyad Abdelkarim

Moayyed Abdelwadood Mesleh

Mohammad Soud Al-Nsoor

Obada Yahia Hussien

**Supervised By**

Prof. Albara Awajan

This project is submitted in partial fulfillment of the requirements for the bachelor’s degree in Artificial Intelligence and Robotics

**Faculty of Artificial Intelligence**

**Al-Balqa Applied University**

**Al-Salt- Jordan**

**February 2024**

# Dedication

This work is sincerely dedicated to our parents, and our faithful friends for their encouragement, and above all, to Almighty Allah, thank you for guiding us and granting us this opportunity and strength to complete our work.

# Acknowledgments

We wish to express our gratitude to our professor, the director of the Center for Innovation, Creativity, and Entrepreneurship, Professor Albara Awajan, being our project's mentor, for his knowledge, motivation, and patience; this project wouldn't have been where it is without his assistance and guidance. We sincerely thank the faculty of artificial intelligence at Al-Balqa Applied University for the opportunity to pursue a career in artificial intelligence.

# Table of Contents

# List of Figures

# List of Tables

# Abstract

Significant advancements have been made in the field of artificial intelligence, particularly in text-to-image models. These models have shown an impressive ability to generate high-quality, varied images from textual prompts. However, they still face challenges in accurately mimicking the visual attributes of subjects from specific reference sets and in generating novel interpretations of these subjects in different contexts. In response to these limitations, this project investigates, implements, and uses one of the novel methods for 'personalizing' text-to-image diffusion models, that may enhance their adaptability and accuracy. In this project, the implemented Generating Images Using Textual Inversion based on the text-to-image diffusion model that involves fine-tuning a pre-trained text-to-image diffusion model using four sets of images to depict four different concepts (styles) oil paintings, dark and scary, cyberpunk game, and line art, as a result, the originally pre-trained model when fine-tuned, associates a text for each of the four mentioned concepts, and accordingly, the fine-tuned model generates high quality, fresh, photorealistic images of the four concepts in various scenes.

# Chapter 1: Introduction

## 1.1 Overview

The pursuit of artistic expression and creativity has long been intertwined with the development of computational models and algorithms. One notable advancement in this realm is Textual Inversion [1] which allows individuals to infuse their unique *style* into generated *content*. By harnessing the power of Textual Inversion, individuals can create visually captivating and stylistically consistent outputs that reflect their artistic preferences.

Textual Inversion [1] is an innovative technique that enables the extraction of novel concepts from a limited set of example images. Initially demonstrated using a latent diffusion model (LDM), this technique has shown promising results. The concepts learned through Textual Inversion can significantly enhance the control and customization of generated images in text-to-image pipelines. By acquiring new "words" in the embedding space of the text encoder, these concepts can be seamlessly integrated into text prompts, resulting in personalized and highly tailored image generation.

Text-to-image models offer unprecedented freedom to guide creation through natural language. It can be exercised to generate images of specific unique concepts, modify their appearance, or compose them in new roles and novel scenes. In other words, how can we use language-guided models to turn some content image into a painting with a specific concept, or a new product based on the style of favorite, or even generating new paints, new shapes, and new creative painting ideas all based on predefined styles.

## 1.2 Project Objectives and Aims

Generative models [1] have made significant strides in recent years, particularly in their ability to create realistic and diverse outputs. However, their lack of a 'human touch' often results in content that doesn't fully align with human preferences, emotions, and needs.

This project focuses on implementing a humanized generative model that aims to better understand and cater to human sensibilities by integrating some human considerations:

1- To implement a stable diffusion model.

2- To master the pre-training of the stable diffusion model.

3- To fine-tune the pre-trained stable diffusion model to create a generating image using the textual inversion model, the fine-tuning process uses a few images of four different concepts, and to learn the model to represent a new "word" in the embedding space of a frozen text-to-image model for each of the four concepts. i.e. fine-tuning the pre-trained model with personal signature examples – a few specific examples enable the model to learn to generate content that aligns more closely with the personal signature's characteristics.

4- Run the implemented generating image using the textual inversion model to generate content images with the desired styles – concepts. This can be used to improve artistic creativity and to enable artists to infuse their unique styles into generated content to create visually captivating and stylistically consistent outputs that reflect their artistic preferences keeping their touch and their signature united with their generated art.

5- Help reduce bias by Controlling bias in generative models, which also will contribute to more accurate and diverse portrayals, promote positive social impact, and ensure that underrepresented groups are not marginalized in the generated content. This will also improve user experience so users can feel more represented and engaged with their generated outputs.

## 1.3 Outline of the Project Report

The rest of the project documentation is organized as follows:

**Chapter 2: Related Work**

This chapter presents concepts and methods related to our project such as image composition, Generative Adversarial Network (GAN), and Style Generative Adversarial Network (StyleGAN).

### Chapter 3: Proposed Work / Methodology

This chapter presents the main algorithms used in our project such as LDM, text embeddings, textual inversion, and evaluation.

### Chapter 4: Project Design & Implementation

This chapter discusses the implementation of the project, clarifies the training of the textual inversion model, the implementation steps, and results & their related discussion.

### Chapter 5: Conclusion and Future Work

This chapter concludes the project and suggests future enhancements and outlooks.

# Chapter Two: Related Work

In [2], a deep image compositing model is proposed in which new images are composed by combining regions from different images, the process of generation of high-quality composites usually involves steps such as segmentation, and foreground color decontamination, this process is a time-consuming, and it degrades the quality of composite images. Moreover, one of the main drawbacks of this approach is that the foreground object comes from a different scene than the background and it is not likely to match the background scene in a way that negatively affects the realism of the composite.

In [3], the first generation of (GANs) is proposed as an artificial intelligence algorithm that addresses the generative modeling problem. GAN involves training a generator network and a discriminator deep neural networks, where the generator produces synthetic data examples, while the discriminator is to differentiate between real and fake examples, through the process of GAN training, the generator progressively enhances its output examples to become more realistic, while the discriminator strives to become more proficient in distinguishing real from fake examples, through iterative training and refinement of these deep neural networks, GANs acquire the ability to generate super quality examples.

In [4], a Deep Convolutional Generative Adversarial Network (DCGAN) is proposed, it is a well-known GAN that uses deep-convolutional-neural-networks (CNNs) for Generator and Discriminator networks, it is proven to be efficient in generating high-quality and realistic images.

In [5] Wasserstein Generative Adversarial Network (WGAN) is designed to overcome certain drawbacks associated with traditional GANs, specifically associated with training stability and mode collapse. By leveraging the Wasserstein distance, WGANs aim to enhance the training process and to achieve more stable and realistic results.

In [6], Progressive GAN is proposed to surpass GANs in terms of stability and image quality. By gradually growing the networks and incrementally increasing the resolution, Progressive GANs have achieved higher quality and more stable image generation compared to previous GAN variants.

In [7], StyleGAN is proposed to emphasize the production of high-quality images, to offer fine-grained control over various aspects of an image's style through the incorporation of style-based techniques.

In [8], Denoising Diffusion Probabilistic Model (DDPM) is proposed to generate high-quality samples that rival the performance of other GANs. DDPMs operate by gradually learning to reverse a diffusion process, which transforms data into a Gaussian distribution over a series of steps. One of the key strengths of DDPMs lies in their structural design, which involves a sequence of denoising steps that incrementally reconstruct data from noise. This method is effective in generating detailed and coherent images and it exhibits versatility in a variety of tasks including image super-resolution and inpainting.
Diffusion models [8], [9]commonly work directly with pixel-level data, but the optimization process for advanced diffusion models can be time-consuming, often requiring hundreds of Graphical Processing Unit (GPU) days. Additionally, the inference process is costly due to the need for sequential evaluations. To make Diffusion Model training feasible on constrained computational resources without compromising quality and versatility, Rombach in [10] implemented them in the latent space of highly capable pre-trained autoencoders [8]. This novel approach, unlike previous methods, enables achieving a nearly optimal balance between reducing complexity and preserving fine details, resulting in significantly enhanced visual fidelity.

Text-to-image generation [11], [12] represents a groundbreaking advancement in the field of artificial intelligence, particularly in computer vision and natural language processing. This technology bridges the gap between textual descriptions and visual imagery, enabling the creation of detailed, realistic images from simple text inputs. These models are constructed using a variety of architectures to encode text, including Recurrent Neural Networks (RNNs),[13], Long Short-Term Memory (LSTM) [13], Transformers, and Bidirectional Encoder Representations from Transformers (BERT) [14].
For image generation, these models typically employ techniques like GANs or diffusion models. GANs, comprising a generator and a discriminator, work in tandem to produce images. The generator creates images based on text descriptions, while the discriminator evaluates these images against real ones, ensuring the generated images are increasingly realistic. This adversarial process results in high-quality, authentic-looking images that closely align with the given text prompts.

Diffusion models, on the other hand, adopt a different approach. They start with a random distribution of pixels and gradually structure these pixels to form coherent images that match the text descriptions. This is achieved by iteratively applying a reverse diffusion process, which involves learning to denoise images, effectively transforming noise into a structured image that reflects the text input. Figure 2.1 shows a simple overview of the text-to-image generation process, which is explored in more depth in Chapter 3.

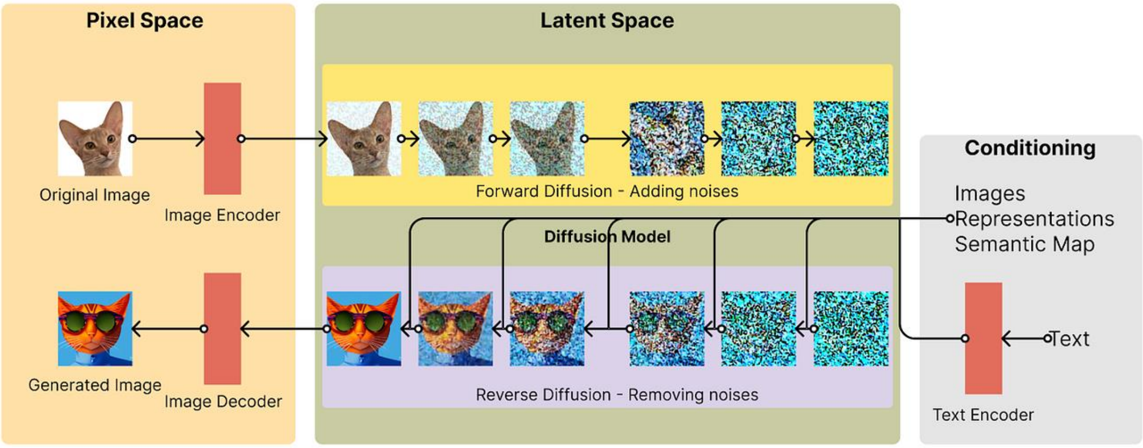


*Figure 2.1: Text-to-Image in diffusion process*[1]

In this project, our focus for image generation lies in diffusion models and textual inversion.

Table 2.1 shows the gap table of generative models from 2014 to 2021.

[1] How Stable Diffusion works, explained for non-technical people by Guodong (Troy) Zhao: https://bootcamp.uxdesign.cc/how-stable-diffusion-works-explained-for-non-technical-people-be6aa674fa1d

## Table 2.1: Gap table of generative models

| model name | Algorithm used | outcomes |
|---|---|---|
| Image compositing [2] | Image compositing model creates new images by combining regions from different images. | It is time-consuming and can degrade image quality. A significant drawback is that the foreground object often comes from a different scene, negatively impacting the realism of the composite by mismatching with the background scene. |
| GANs [3] | Consist of two parts, the generator that generate a new image, and a discriminator that differentiate between real and fake examples. | GANs can generate high-quality synthetic examples, yet they are limited by image resolution. |
| DCGAN [4] | It is GAN that uses deep Convolutional Neural Networks. | This GAN has been proven to be efficient in generating high-quality and realistic images, and it offers more stable training. However, it is limited by image resolution. |
| WGAN [5] | It is designed to address drawbacks in traditional GANs, such as training stability and mode collapse. The model uses the Wasserstein distance to improve training by assessing the dissimilarity between real and generated data distributions. | The model achieves more stable and realistic outcomes and uses a meaningful loss metric. However, it incurs increased computational costs. |
| Progressive GAN [6] | This model aims to surpass traditional GANs in stability and image quality by gradually growing networks and incrementally increasing resolution. | This approach has resulted in the generation of higher-quality and more stable images compared to earlier GAN variants. However, it requires more training time. |
| StyleGAN [7] | It is designed to prioritize the generation of high-quality images by incorporating style-based techniques, it aims to provide fine-grained control over various aspects of an image's style. | The model’s ability to produce refined and customizable outputs, but it is challenged by high computational complexity and limited image diversity. |
| Diffusion Model [8] | It is a deep learning technique that generates high-quality images by gradually transforming noise into structured visual data. | Generation of high-quality, detailed images, capability to produce diverse and complex visual content, improved robustness in image generation compared to some traditional methods. However it is complex, and requires high computational power. |
| Latent Diffusion Model [10] | It operates in a compressed, latent space to efficiently generate high-quality images with lower computational requirements compared to traditional diffusion models. | The LDM is effective for generating high-quality and coherent images, with versatility in tasks such as image super-resolution and inpainting. |

# Chapter Three: Methodology

## 3.1 Latent Diffusion Models

Diffusion Models [8] are a type of probabilistic model designed to understand and capture the underlying data distribution, denoted as $p(x)$ by iteratively removing noise from a variable that is normally distributed. Latent Diffusion Models (LDMs) [15] are enhanced version of diffusion models, which fall under the recently introduced category of Denoising Diffusion Probabilistic Models (DDPMs), noting that LDMs operate specifically within the latent space of an autoencoder. LDMs are comprised of two main components. Firstly, the Autoencoder, pre-trained on a vast set of images, includes an encoder $E$ that maps images $x$ (belonging to dataset $D_x$) into a Spatial Latent Code $z = E(x)$. This process involves regularization using either KL-divergence loss or Vector-Quantization [16]. The decoder D then maps the latent representation back such that $D(E(x)) \approx x$. Secondly, the diffusion model is trained to produce codes within this learned latent space. It includes a function $C_\theta(y)$ that maps a conditioning input $y$ into a conditioning vector. The LDM loss $L_{LDM}$ is then given by the following Eq.3.1:

$$L_{LDM} := E_{z\sim\varepsilon(x),y,\epsilon\sim N(0,1),t}[||\epsilon - \epsilon_\theta(z_t, t, C_\theta(y))||_2^2] \qquad \text{Eq.3.1}$$

where t represents time steps, $\mathrm{Z}_t$ is the Latent Noise up to time t, $\varepsilon$ is an unscaled noise sample, and $\varepsilon_\theta$ is the denoising network. The objective is to remove the noise added to the Latent Representation of an image. During training, $C_\theta$and $\varepsilon_\theta$optimized to minimize the LDM loss. Concurrently, a random noise tensor is sampled and iteratively denoised to produce a new image latent, $Z_0$. This latent code is then transformed back into an image using the pre-trained decoder, resulting in the reconstructed image $X' = D(Z_0)$ (see Figure 3.1).

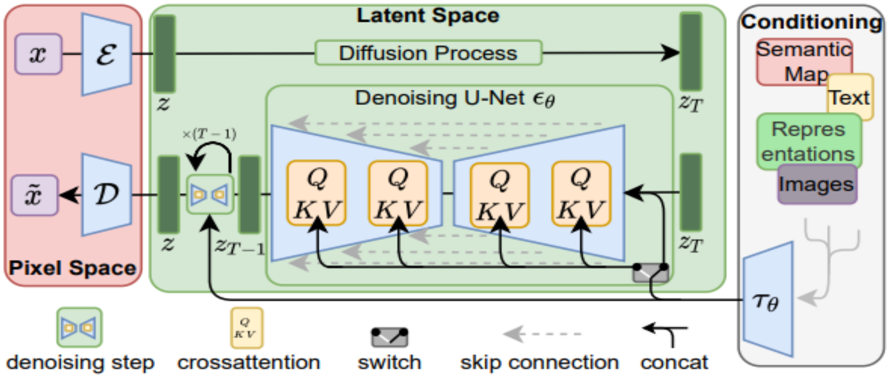


*Figure 3.1: LDMs [15]*

The publicly available 1.4 billion parameter text-to-image model [15] was pre-trained on the LAION-5B [17] dataset, and it is used in this project. In this model, $C_\theta$ is implemented using BERT text encoder, with $y$ representing a text prompt.

## 3.2 Text Embeddings

Embeddings [18] are not understandable by humans since they are vectors of numbers inferred by the computer. In image embedding, each number holds one property of the image. To illustrate image embeddings, one may say, that the first number tells how probable the house is in the image, the second might say that the image contains clouds, and the last one gives information on the presence of the sun. On the other hand, text embedding is to relate the tokens to each other after the vectorization of the word, embedding means to encode the meaning of the word in vector space to be understandable by the computer.

As discussed in [1], common text encoder models, like BERT, adhere to a standard text processing procedure. Initially, each word or sub-word in the input text is converted into a token, represented by an index from a predefined dictionary. Each token is then associated with a unique embedding vector, accessible through an index-based lookup.

The learning process of the text encoder often involves the training of these embedding vectors to improve its overall performance. In this project, Rinon Gal's method is adopted for choosing the embedding space as the target for inversion [1]. Specifically, a new string, $S*$, represents the novel concept we aim to learn. By intervening in the embedding process, we gain the ability to replace the vector associated with the tokenized string with a new embedding, $V*$. This process effectively 'injects' the new concept into our vocabulary. Consequently, we can seamlessly construct sentences that include this concept, treating it as we would any standard word.

## 3.3 Textual inversion

These novel embeddings are discovered by relying on a limited collection of images (usually 3-8 in this project) that depict the desired concept in diverse scenarios, including various backgrounds or poses. The optimal value of $V *$ is determined through direct optimization by minimizing the LDM loss, as outlined in Eq.3.1, using images randomly selected from this small set.

To influence the generation process, randomly selected neutral context texts are utilized (see Figure 3.2).. These texts are sourced from the Contrastive Language-Image Pre-Training (CLIP) ImageNet templates [19] and consist of prompts structured as 'A photo of $S *$', 'A rendition of $S *$', and so forth More templates are provided for reference in the implementation chapter.

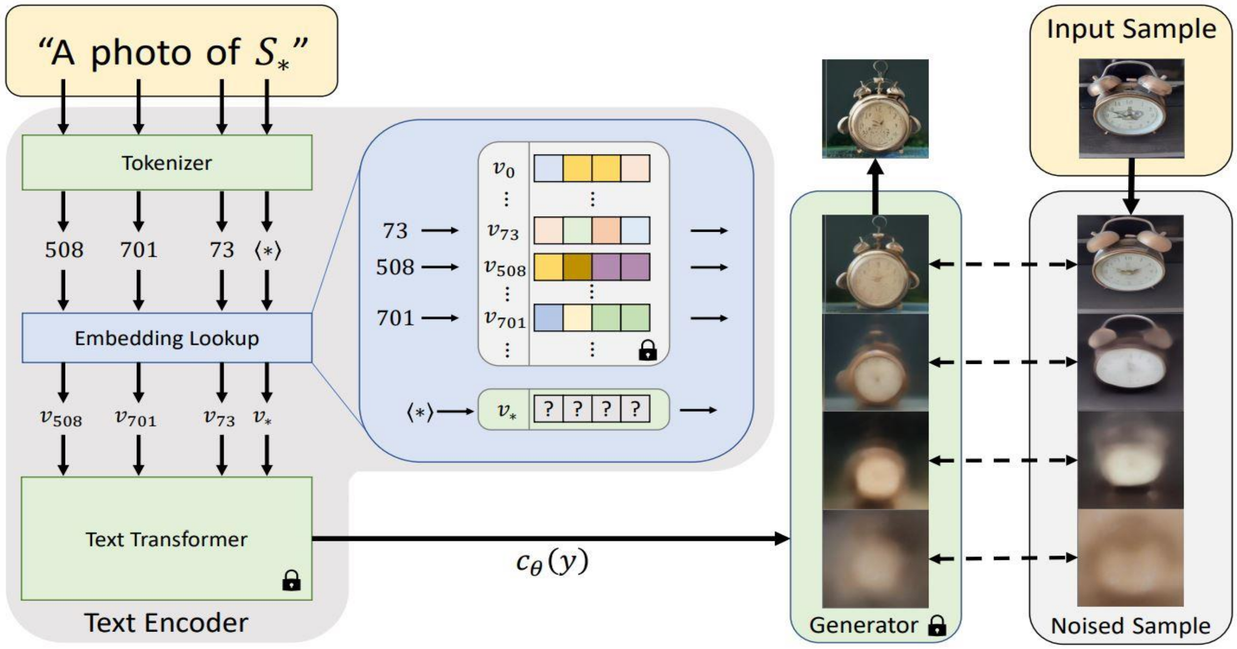


*Figure 3.2: Textual inversion*[2]

[2] Textual inversion at GitHub: https://textual-inversion.github.io/

## 3.4 Evaluation

The model's consistency is evaluated by assessing its performance using human-understandable metrics. The Mean Opinion Score (MOS) [20], a numerical measure ranging from 1 to 5, is used for the human judgment of the overall quality of an event or experience, such as a generated image in our work. Additionally, the Inception Score (IS) [21] is employed, which is an algorithm designed to assess the quality of images produced by a generative image model.

Furthermore, two commonly used metrics in the field of generative models are the Fréchet Inception Distance (FID) [22], which is more effective than the IS in capturing the similarity of generated images to real ones, and CLIP [23], an open-source, multi-modal, zero-shot model; the CLIP model, given an image and text descriptions, predicts the most relevant text description for that image without being optimized for a specific task.

### 3.4.1 CLIP

CLIP, primarily designed to rank image generation based on its similarity to text (indicating whether an image and text are related or not), is a multi-modal model introduced by OpenAI[3] in 2021, trained on 400M [24] pairs of an image-text dataset, it has various applications in the field of machine learning, with our focus being on its use as an evaluation metric in generative models.

The model consists of two sub-models, known as encoders, one being a text encoder that embeds text into a mathematical space and the other an image encoder that embeds images into the same space (see Figure 3.3). During training, high cosine similarity [25]is assigned to corresponding image-text pairs, while a low similarity value is given to disjoint image-text pairs.

[3] OpenAI: https://openai.com/

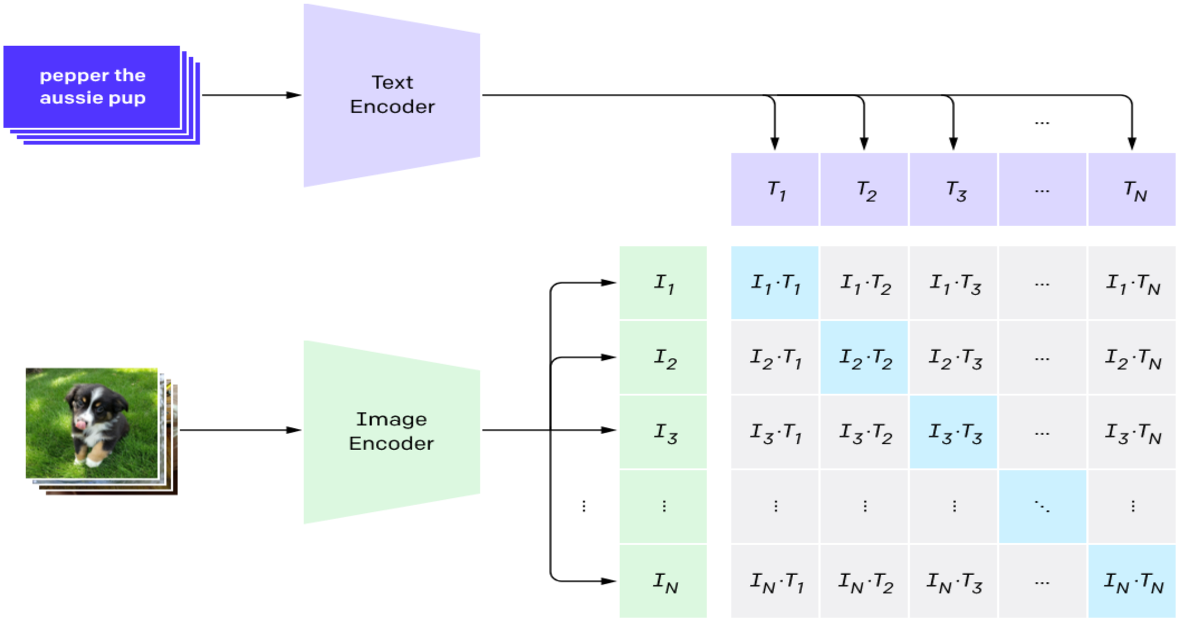

pepper the aussie pup
Text Encoder
Image Encoder
$T_1$ $T_2$ $T_3$ ... $T_N$
$I_1$ $I_2$ $I_3$ ⋮ $I_N$
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

*Figure 3.3: Overview of how CLIP has been trained [23]*

### 3.4.1.1 CLIP in the stable diffusion model:

The model determines (see Figure 3.4) the cosine similarity between the embeddings of the generated image and the input text. If the cosine similarity is high, it indicates that the image accurately represents the text (prompt); conversely, a low similarity suggests that it does not.

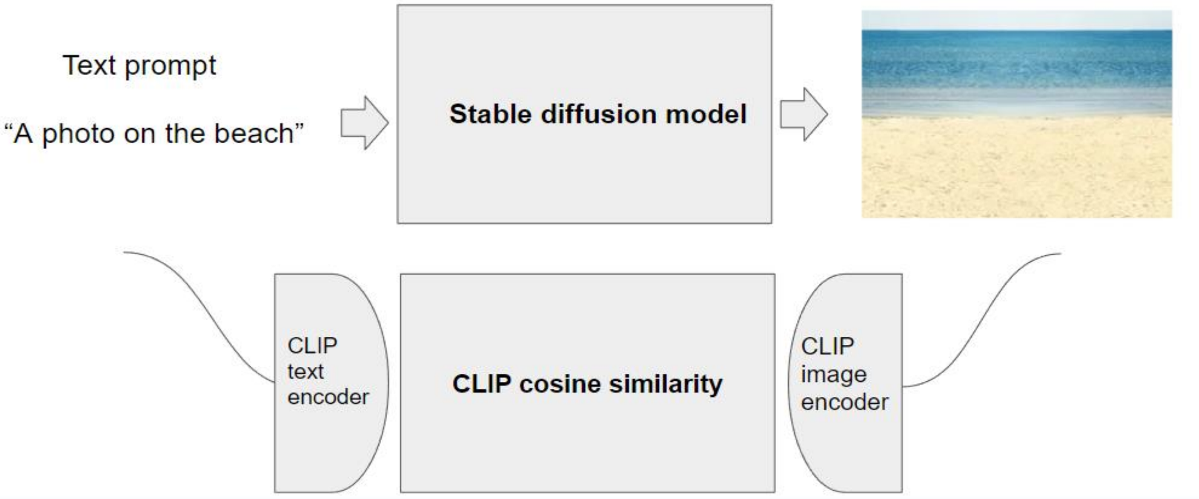


*Figure 3.4: Explanation of clip work in stable diffusion.*

### 3.4.1.2 CLIP in Textual Inversion:

In textual inversion, the similarity is computed between the text and input images on one side, and the generated image on the other, resulting in two values: Clip I and Clip T. This is depicted in the following image (see Figure 3.5):

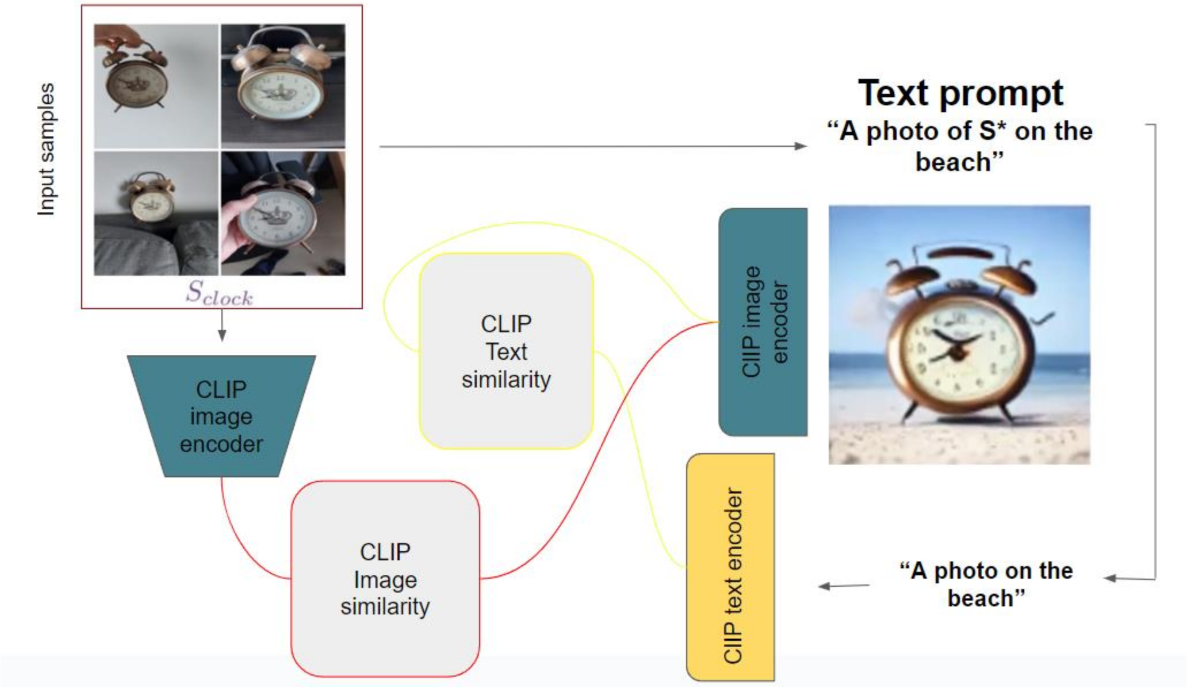


Figure 3.5: We compute the similarity between the input images and the generated image, *and the similarity between the text prompt without the pseudo word and the generated image. If we get high similarity, the generated image represents the inputs of the text and the images (3 to 4 use input images).*

However, CLIP's scoring does not capture other aspects of image quality, such as visual fidelity, realism, or diversity. It focuses more on the semantic alignment between the text and the image. So, the FID [22] metric is used to indicate the quality of generated images.

### 3.4.2 FID

FID [22] is a metric that calculates the distance between feature vectors calculated for real and generated images. The score combines both quality and diversity. The difference between two Gaussians (generated and real-world images) is measured by the Frechet distance also known as the Wasserstein-2 distance. The lower FID score indicates better image quality, per contrast higher FID indicates lower image quality.

# Chapter Four: Project Design & Implementation

## 4.1 Project Design

### 4.1.1 Project Design steps

1- Gather 3 to 5 images that revolve around the concept you want to extract. These images should be representative and provide a clear understanding of the concept.

2- Create a new word that encapsulates your desired concept. This word will be used to augment the pretrained language model and enhance its ability to understand and generate content related to your concept.

3- Perform textual inversion by linking your newly created concept to your new word to the pre-trained language model. This association will enable the model to generate images that correspond to your concept.

4- Compose a new prompt that incorporates your newly defined concept word and input it into the pre-trained language model. Retrieve the output image generated by the model, which will be based on the concept extracted from the initial input images. This output image represents the model's interpretation and visualization of your concept.

### 4.1.2 Project Design Flow Diagram

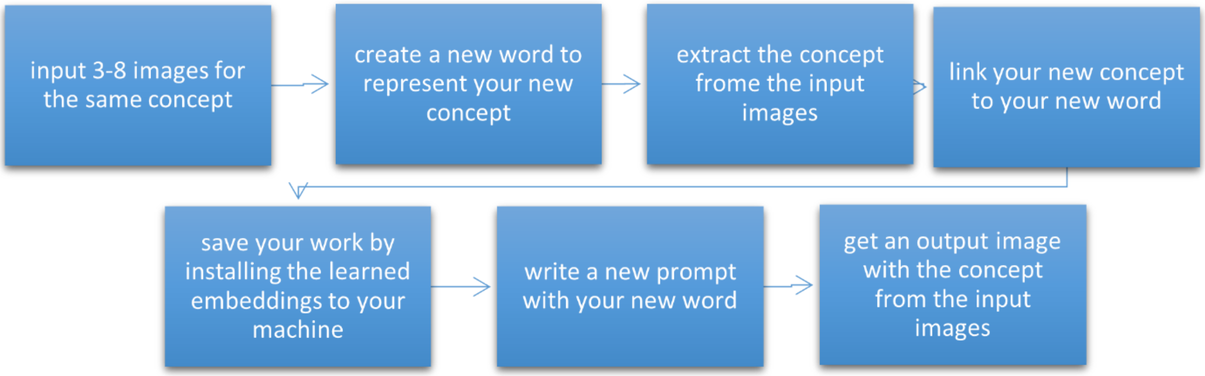

input 3-8 images for the same concept
create a new word to represent your new concept
extract the concept frome the input images
link your new concept to your new word
save your work by installing the learned embeddings to your machine
write a new prompt with your new word
get an output image with the concept from the input images

*Figure 4.01: Project Design flow diagram*

## 4.2 Project Implementation

The Stable Diffusion Model (SDM) is an open-source, pre-trained text-to-image latent diffusion model that has been trained on LAION-5B[4], consisting of 5 billion image-text pairs. Training such a model takes a very long time and requires significant computing capability. Specifically, the model was trained for 150,000 GPU hours using 256 Nvidia A100 GPUs.

Our textual inversion training is based on the Stable Diffusion Model. Due to the lengthy training time required for the Stable Diffusion Model, we could not afford to train our model from scratch to use as the basis for our textual inversion project.

The first section of implementing the Textual Inversion is the setup, which introduces you to the main library used in fine-tuning the Stable Diffusion (Training the Textual Inversion).

### 4.2.1 Setup

The main library used to leverage the pre-trained Stable Diffusion model, along with its sub-libraries like U-net, CLIP, and Transformer, is as follows.

**Diffusers Library**: The Diffusers library, is a Python package developed by Hugging Face, designed for working with diffusion models. The library includes implementations of some of the latest and most advanced diffusion models. It is used for a variety of tasks including image generation, image-to-image translation, super-resolution, and potentially audio synthesis.

**Transformers Library**: The Transformers library [26] , developed by Hugging Face, is a comprehensive and widely used Python library that provides a large collection of state-of-the-art pre-trained models for Natural Language Processing (NLP) and Natural Language Understanding (NLU) tasks.

**Computational Resources**: Ensure access to adequate computational resources. Fine-tuning deep learning models is resource-intensive and typically requires powerful graphical processing units (GPUs). Fine-tuning the SDM on Google Colab[5] with an Nvidia T4 GPU takes two hours for 2,000 steps, whereas on a local machine equipped with an Nvidia RTX 3070[6], it takes only 90 minutes for 10,000 steps.

[4] LAION-5B website: https://laion.ai/blog/laion-5b/

[5] Google colab: https://colab.research.google.com/

[6] Nvidia website: https://www.nvidia.com/en-me/geforce/graphics-cards/30-series/rtx-3070-3070ti/

### 4.2.2 Dataset Preparation

Dataset used for fine-tuning must meet certain constraints to ensure high accuracy and adaptability. In the text-to-image models, the dataset is split into two branches: the image-text pairs and the prompt templates.

#### 4.2.2.1 Image-Text Pairs Dataset

In this project, four sets of images that represent four different concepts (styles) were collected from different Internet sites. These sets should be diverse enough to cover different aspects of each concept used in the project. Then, a textual description is associated with each image, this textual description should be consistent and accurately reflect the new concept.

Consistency means that the description should have a consistent format or structure, this helps the model to better understand and generalize the concept.

Furthermore, we can apply data augmentation techniques like cropping, rotating, scaling, or color adjustment to increase the diversity of the dataset. This can help in making the model more robust to variations.

#### 4.2.2.2 Prompts Dataset

Textual Inversion involves a collection of textual prompts used to train or fine-tune generative models like SDM. These prompts often describe objects or scenes and, a collection of such prompts that paired with images called prompt dataset. During fine-tuning, the model generates outputs based on these prompts. The key is to adjust the parameters of the model to improve how closely its outputs match the provided (ideal) outputs. This iterative process is designed to gradually refine the model's capability to understand and generate content that aligns with the new concept or style being introduced.

The following is a small set of prompts used in our project implementation:

```
style_templates= [
    "a painting in the style of {}",
    "a rendering in the style of {}",
    "a cropped painting in the style of {}",
    "the painting in the style of {}",
    "a clean painting in the style of {}",
    "a dirty painting in the style of {}",
    "a dark painting in the style of {}"]

Object_templates = [
    "a photo of a {}",
    "a rendering of a {}",
    "a cropped photo of the {}",
    "the photo of a {}",
    "a photo of a clean {}",
    "a photo of a dirty {}",
    "a dark photo of the {}"]
```

### 4.2.3 Model Training

The goal of training the Textual Inversion is to introduce a new word (a keyword or pseudo-word) into the model's vocabulary. This word refers to a concept (such as a style or an object) previously unseen by the model. The first step in training the model using our dataset is to determine the name of the token we are going to use, the token should be unique and not in the vocabulary known by the model, so it is not preferred to use words that already exist in the vocabulary list of the model. In this project, new sentences like the following are used: Mj_toy, lines_paint_1, wen_sed. The token is then converted into an embedding and the embedding learning starts.

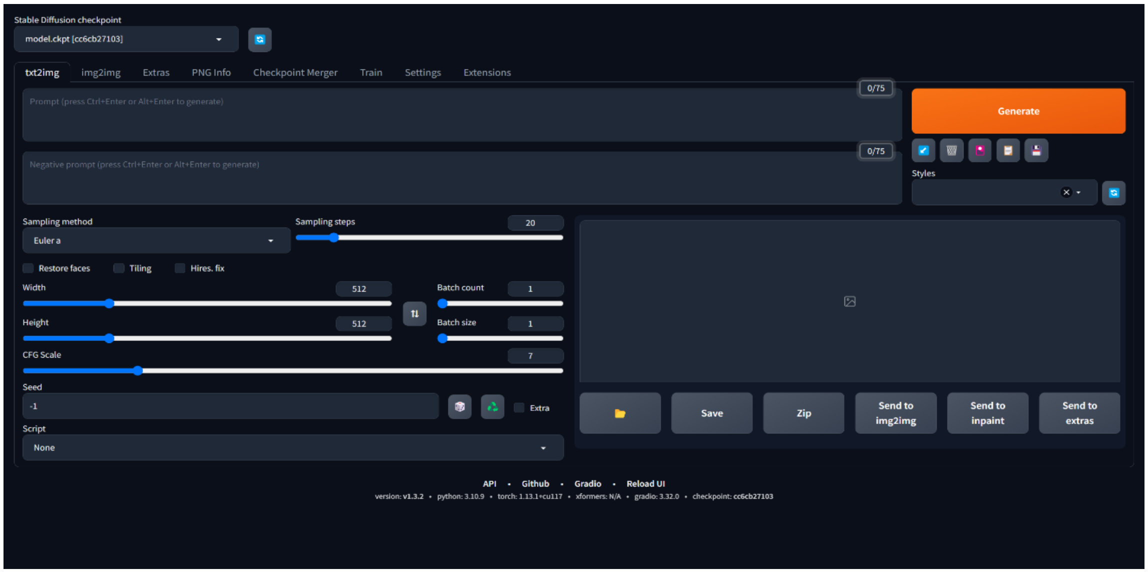
**Embedding Learning**: The objective of fine-tuning the model in Textual Inversion is to create and refine an embedding that represents a specific concept not originally included in the model's training data. This embedding is associated with a unique token, which is used in text prompts to signify a particular concept. During the training process, the embedding is iteratively updated. This is achieved by comparing the outputs generated by the model when using the special token in prompts against the target outputs (such as images) that exemplify the concept. The update is typically guided by a loss function that measures how well the model's outputs align with the concept as represented in the training dataset. This process adjusts the embedding to embody the concept more accurately, allowing the model to generate content closely aligned with the intended idea or style when the token is used in prompts. The mechanism behind the loss functions is a cosine similarity which is done by CLIP pretrained model. Note that the embeddings updating is limited to the concept embedding, the pre-trained model embeddings are kept untouched.

**Learning Rate**: The learning rate used should be appropriate for fine-tuning, it should be less than the learning rate used in the initial training (pre-training). Bigger learning rates cause overfitting of the training data and destroy the initial model. Noting that the learning rate is experimentally set to 0.0005.

**Parameters optimization**: The model's weights are adjusted using optimization algorithms (Adaptive moment estimation - Adam) to minimize the loss function, which measures the discrepancy between the model's output and the target concept. Adam is used to minimize the training cost.

## 4.2.4 Web User Interface

The web user interface[7] hosted in a local machine is implemented using the following steps as shown in Figure 4.1:



Stable Diffusion checkpoint: model.ckpt [cc6cb27103]

Prompt (press Ctrl+Enter or Alt+Enter to generate) 0/75

Negative prompt (press Ctrl+Enter or Alt+Enter to generate) 0/75

Sampling method: Euler a

Sampling steps: 20

□ Restore faces □ Tiling □ Hires. fix

Width: 512

Height: 512

Batch count: 1

Batch size: 1

CFG Scale: 7

Seed: -1 □ Extra

Script: None

version: v1.3.2 • python: 3.10.9 • torch: 1.13.1+cu117 • xformers: N/A • gradio: 3.32.0 • checkpoint: cc6cb27103

*Figure 4.02: Web User Interface*

1- Clone the GitHub repository to download the web interface.

2- Store the pre-trained SDM in the corresponding directory of the web interface (models directory).

[7] Web UI: https://github.com/AUTOMATIC1111/stable-diffusion-webui

3- Determine the SDM as a model checkpoint.

4- Fine-tune the SDM using the training dataset after creating a new embedding from the new embedding subtab.

It is noted that the learning embedding is stored in the embedding's directory on the user's computer, and the user can use the model by inserting a text containing the token initialized in the text-to-image tab on the user interface.

When using the web User Interface, users need to pay attention to following terms and definitions:

- **Stable Diffusion checkpoint**: Stable Diffusion Models, or checkpoint models, are pre-trained Stable Diffusion weights for generating a particular style of images. What kind of images a model generates depends on the training images.
- **Prompt**: Prompt is the text-based instructions on which the output pictures are designed based on.
- **Negative Prompt**: a way to use Stable Diffusion in a way that allows the user to specify what he doesn't want to see.
- **Sampling method**: Sampling is the denoising process and there are various methods for that.
- **Sampling steps**: The number of iterations Stable Diffusion runs to go from random noise to a recognizable image.
- **Image size**: width and height.
- **CFG Scale**: CFG stands for Classifier Free Guidance scale. Which is a parameter that controls Stable Diffusion and how 'strict' it should follow the prompt input in image generation.
- **Seed**: A seed is a number from which Stable Diffusion generates noise.

### 4.2.5 Examples

The Textual Inversion model is trained using four different datasets (The image size of each image is 512x512) and achieved high accuracy.

**Oil painting style dataset**: The following images are used to fine-tune the SDM and to extract the style common among them (see Figure 4.2). The token used to save this style is '<oil_paint>'. Noting that the model trained for 2,000 steps. Figure 4.3 shows a set of generated images based on the oil painting training dataset:

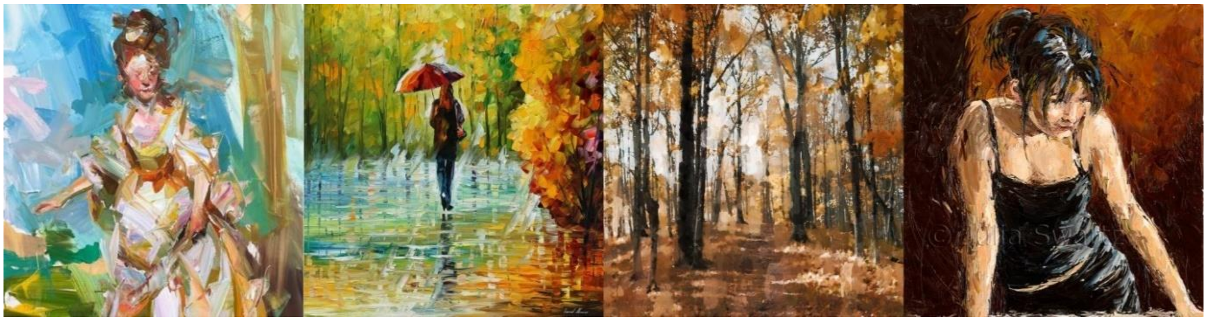
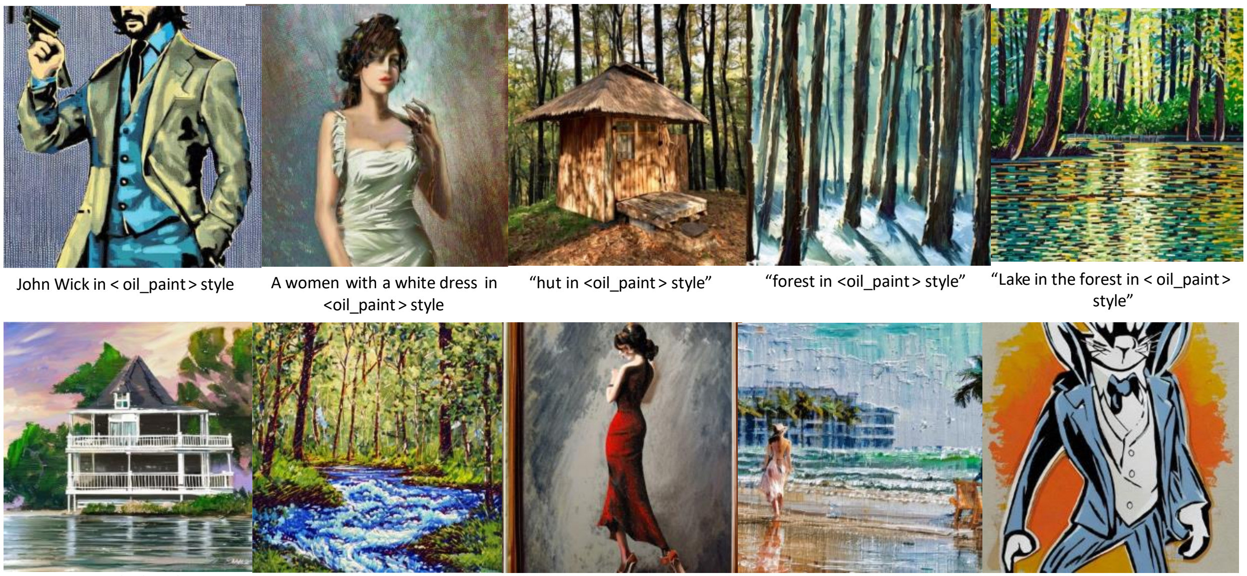


*Figure 4.03: Oil painting training dataset*



John Wick in < oil_paint> style

A women with a white dress in <oil_paint> style

“hut in <oil_paint> style”

“forest in <oil_paint> style”

“Lake in the forest in < oil_paint> style”

House on the beach in <oil_paint> style.

Green river in < oil_paint> style.

“classy women in < oil_paint> Style”

“hotel on the beach in <oil_paint> style”

“Bugs Bunny in < oil_paint> style”

*Figure 4.04: Generated example using the oil painting training dataset*

**Cyberpunk 2077 game style dataset**: The Cyberpunk game features a distinctive style in its visual scenes. This style dataset (see Figure 4.05) is used to fine-tune the SDM and generate art images in the same aesthetic, as shown in Figure 4.06.

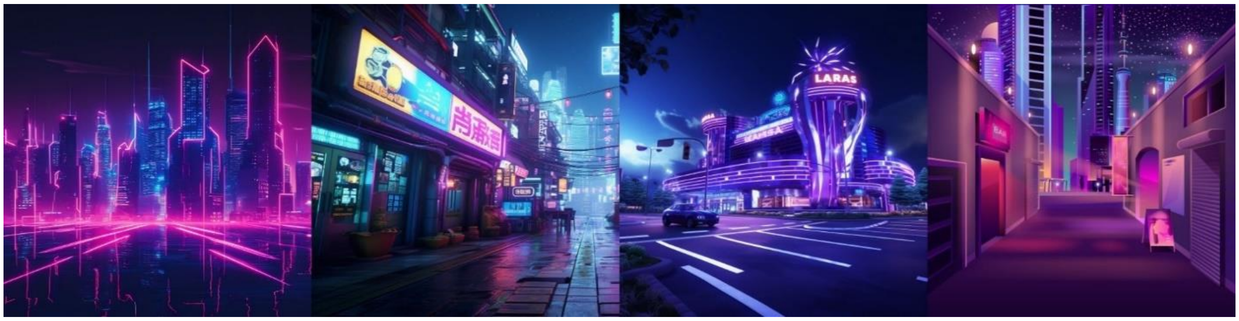
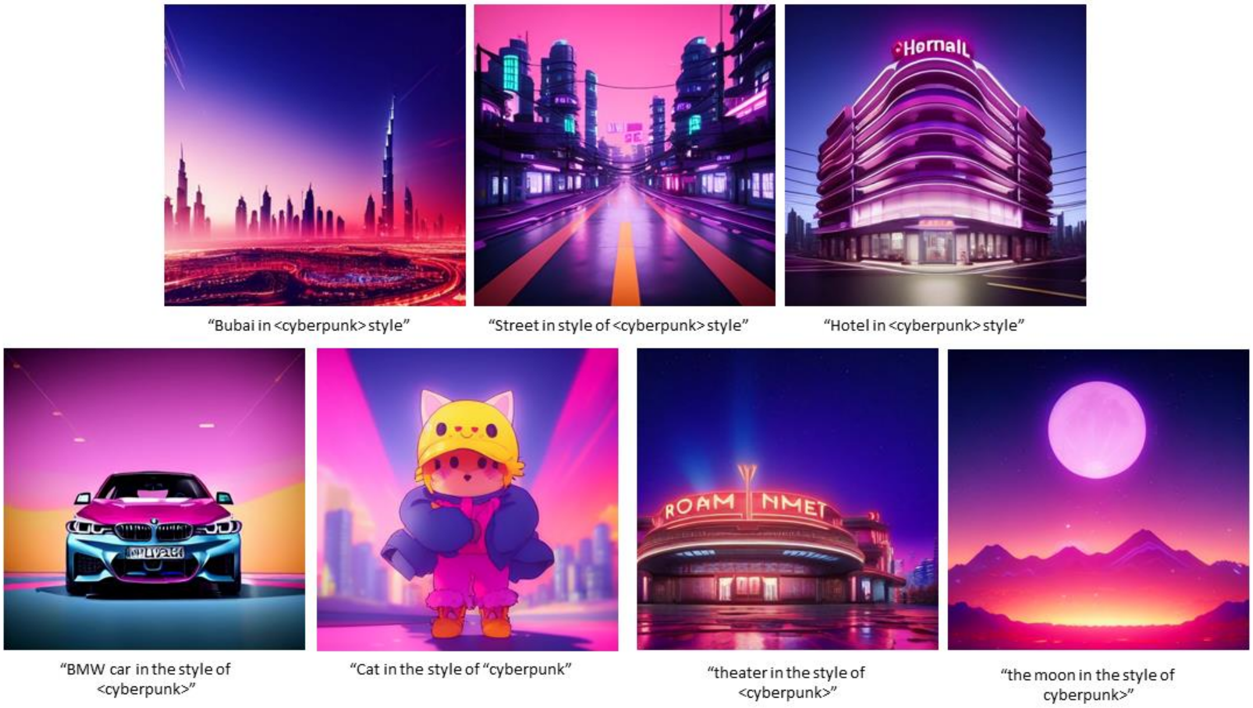


*Figure 4.05: Cyberpunk training dataset*



“Bubai in <cyberpunk> style”

“Street in style of <cyberpunk> style”

“Hotel in <cyberpunk> style”

“BMW car in the style of <cyberpunk>”

“Cat in the style of “cyberpunk”

“theater in the style of <cyberpunk>”

“the moon in the style of cyberpunk>”

*Figure 4.06: Generated example using the Cyberpunk game training dataset*

**Lines Painting style dataset**: Line art uses the lines to create 3D images, this style images (see Figure 4.07) to fine-tune the SDM and generate art images (see Figure 4.08).

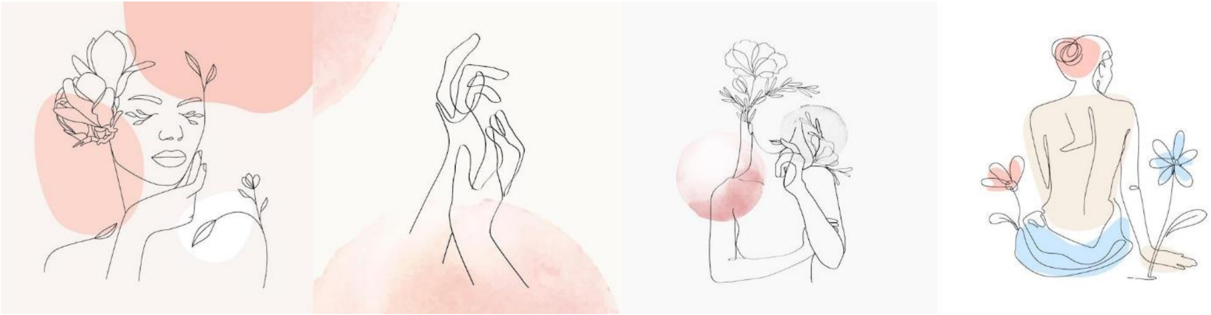
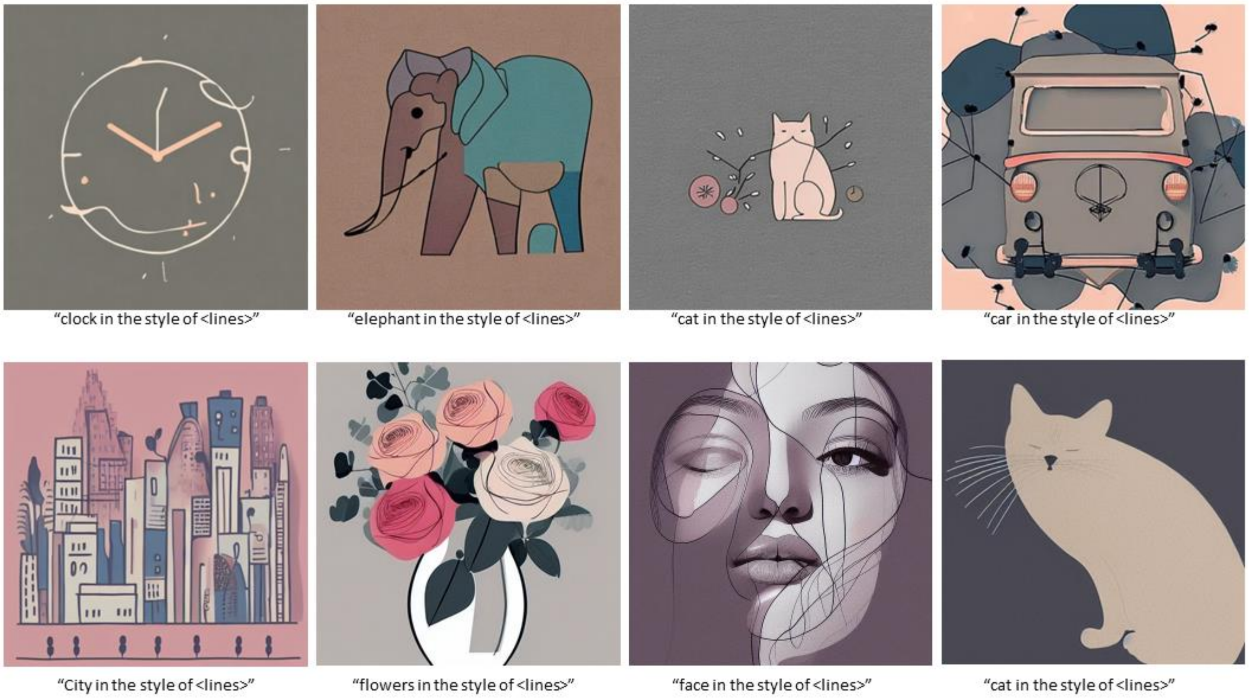


*Figure 4.07: Lines art training dataset*



"clock in the style of <lines>"

"elephant in the style of <lines>"

"cat in the style of <lines>"

"car in the style of <lines>"

"City in the style of <lines>"

"flowers in the style of <lines>"

"face in the style of <lines>"

"cat in the style of <lines>"

*Figure 4.08: Generated example using the lines art training dataset*

**Dark and Scary style dataset**: this dataset (see Figure 4.09) contains dark images and scary landmarks, is used to train the SDM as shown in (Figure 4.10).

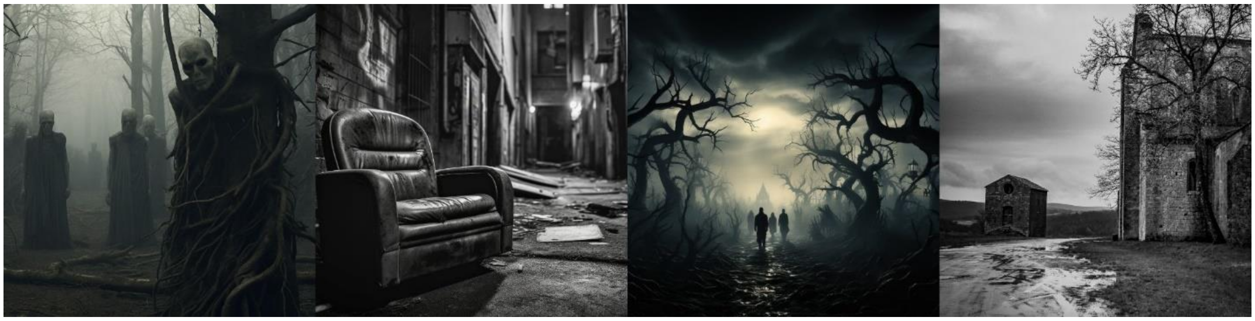
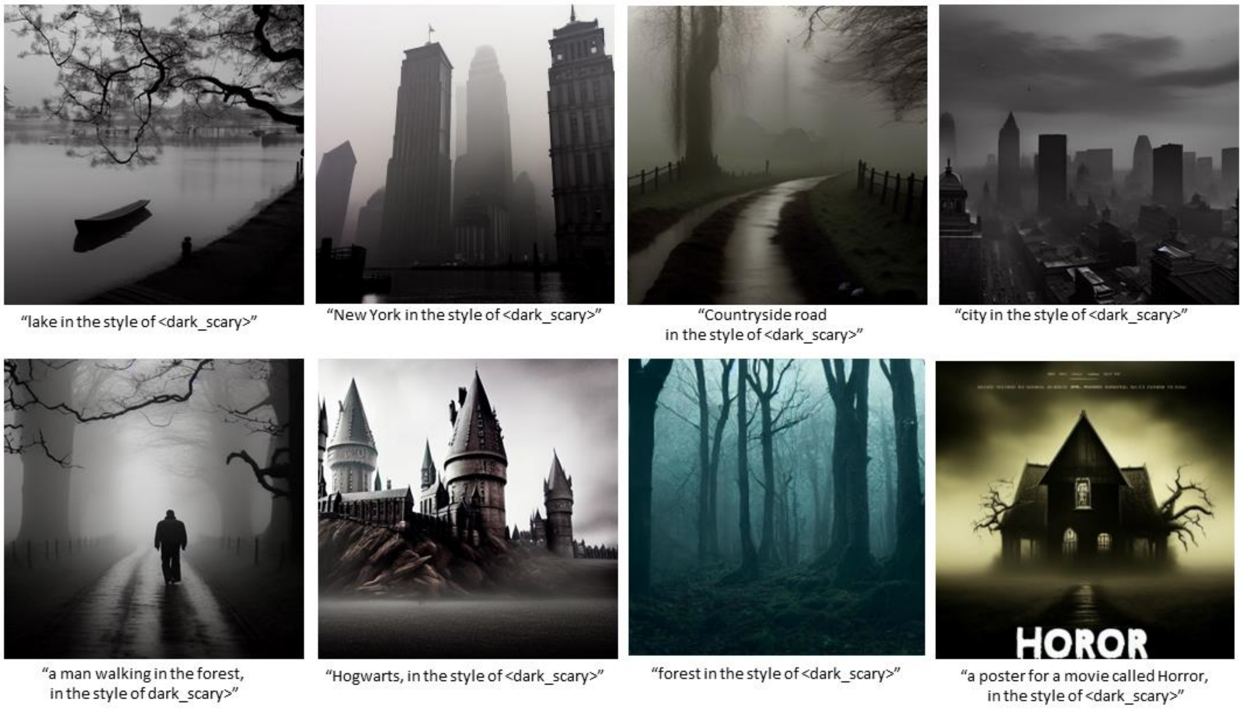


*Figure 4.09: Dark and scary training dataset*



"lake in the style of <dark_scary>"

"New York in the style of <dark_scary>"

"Countryside road
in the style of <dark_scary>"

"city in the style of <dark_scary>"

"a man walking in the forest,
in the style of dark_scary>"

"Hogwarts, in the style of <dark_scary>"

"forest in the style of <dark_scary>"

"a poster for a movie called Horror,
in the style of <dark_scary>"

*Figure 4.10: Generated example using the dark and scary training dataset*

# Chapter Five: Conclusion and Future Work

## 5.1 Conclusion

This project implements the concept of personalized, language-driven generation. This involves utilizing a text-to-image model to generate images that embody specific concepts in new environments and scenes. The process, known as Textual Inversions, works by transforming these concepts into novel pseudo-words within the textual embedding space of an already trained text-to-image model. These pseudo-words can then be seamlessly integrated into various scenes through straightforward natural language descriptions, facilitating easy and intuitive adjustments. Essentially, this method enables users to employ multi-modal information — it simplifies editing through a text-based interface while using visual indicators to signal when the boundaries of natural language are being approached. A Textual Inversion on LDM is implemented in this project, this implementation is based on one of the largest text-to-image model available to the public – LDM, moreover, this novel generative model is investigated, implemented to generate high quality images for different styles.

. In these models, aspects like text-to-image alignment, consistency in shape, and the fidelity of image generation could see further enhancements.

## 5.2 Future Work

In future, a friendly website for this project is to be implemented and publicly used to enable practitioners and students to practice image generation model. Moreover, more sophisticated models will be served in that future website, this future image generation model is expected to be served with a low-cost monthly subscription. The proposed web service will allow users to effortlessly create custom images tailored to their specific needs and preferences. By integrating Textual Inversion with the SDM, subscribers can input simple text prompts to generate high-quality, unique images. Whether it's for professional graphic design, content creation, or personal projects, our platform will provide an intuitive interface that simplifies the process of transforming textual descriptions into vivid visual representations. The subscription model ensures continuous access to the latest advancements in AI-driven image generation, making it an invaluable tool for creatives and businesses looking to stay at the forefront of digital imagery and design.

# REFERENCES

[1] R. Gal *et al.*, “An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion,” Aug. 2022, [Online]. Available: http://arxiv.org/abs/2208.01618

[2] H. Zhang, J. Zhang, F. Perazzi, Z. Lin, and V. M. Patel, “Deep Image Compositing.”

[3] I. J. Goodfellow *et al.*, “Generative Adversarial Networks,” Jun. 2014, [Online]. Available: http://arxiv.org/abs/1406.2661

[4] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” Nov. 2015, [Online]. Available: http://arxiv.org/abs/1511.06434

[5] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein Generative Adversarial Networks,” 2017.

[6] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive Growing of GANs for Improved Quality, Stability, and Variation,” Oct. 2017, [Online]. Available: http://arxiv.org/abs/1710.10196

[7] T. Karras NVIDIA and S. Laine NVIDIA, “A Style-Based Generator Architecture for Generative Adversarial Networks Timo Aila NVIDIA.” [Online]. Available: https://github.com/NVlabs/stylegan

[8] J. Ho, A. Jain, and P. Abbeel, “Denoising Diffusion Probabilistic Models,” Jun. 2020, [Online]. Available: http://arxiv.org/abs/2006.11239

[9] P. Dhariwal and A. Nichol, “Diffusion Models Beat GANs on Image Synthesis,” May 2021, [Online]. Available: http://arxiv.org/abs/2105.05233

[10] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-Resolution Image Synthesis with Latent Diffusion Models,” Dec. 2021, [Online]. Available: http://arxiv.org/abs/2112.10752

[11] C. Saharia *et al.*, “Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding,” May 2022, [Online]. Available: http://arxiv.org/abs/2205.11487

[12] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical Text-Conditional Image Generation with CLIP Latents,” Apr. 2022, [Online]. Available: http://arxiv.org/abs/2204.06125

[13] R. C. Staudemeyer and E. R. Morris, “Understanding LSTM -- a tutorial into Long Short-Term Memory Recurrent Neural Networks,” Sep. 2019, [Online]. Available: http://arxiv.org/abs/1909.09586

[14] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.04805

[15] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-Resolution Image Synthesis with Latent Diffusion Models,” Dec. 2021, [Online]. Available: http://arxiv.org/abs/2112.10752

[16] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, “Neural Discrete Representation Learning,” Nov. 2017, [Online]. Available: http://arxiv.org/abs/1711.00937

[17] C. Schuhmann *et al.*, “LAION-5B: An open large-scale dataset for training next generation image-text models,” Oct. 2022, [Online]. Available: http://arxiv.org/abs/2210.08402

[18] A. Mandelbaum and A. Shalev, “Word Embeddings and Their Use In Sentence Classification Tasks,” Oct. 2016, [Online]. Available: http://arxiv.org/abs/1610.08229

[19] A. Radford *et al.*, “Learning Transferable Visual Models From Natural Language Supervision”, Accessed: May 24, 2023. [Online]. Available: https://github.com/OpenAI/CLIP.

[20] Y. Leng, X. Tan, S. Zhao, F. Soong, X.-Y. Li, and T. Qin, “MBNet: MOS Prediction for Synthesized Speech with Mean-Bias Network,” Feb. 2021, [Online]. Available: http://arxiv.org/abs/2103.00110

[21] S. Barratt and R. Sharma, “A Note on the Inception Score,” Jan. 2018, [Online]. Available: http://arxiv.org/abs/1801.01973

[22] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium,” Jun. 2017, [Online]. Available: http://arxiv.org/abs/1706.08500

[23] A. Radford *et al.*, “Learning Transferable Visual Models From Natural Language Supervision,” Feb. 2021, [Online]. Available: http://arxiv.org/abs/2103.00020

[24] C. Schuhmann *et al.*, “LAION-400M: Open Dataset of CLIP-Filtered 400 Million Image-Text Pairs,” Nov. 2021, [Online]. Available: http://arxiv.org/abs/2111.02114

[25] L. Metcalf and W. Casey, “Chapter 2 - Metrics, similarity, and sets,” in *Cybersecurity and Applied Mathematics*, L. Metcalf and W. Casey, Eds., Boston: Syngress, 2016, pp. 3–22. doi: https://doi.org/10.1016/B978-0-12-804452-0.00002-6.

[26] T. Wolf *et al.*, “HuggingFace’s Transformers: State-of-the-art Natural Language Processing,” Oct. 2019, [Online]. Available: http://arxiv.org/abs/1910.03771