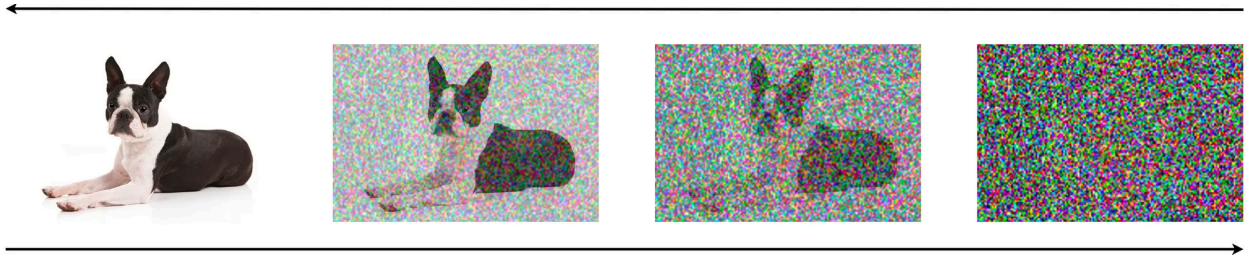
Text to Image Generator using StableDiffusion Model

Problem Statement:
Creating a text to image generator with a stable diffusion model seeks to address challenges in translating text into coherent and realistic visual representations.

Stable Diffusion:
Stable Diffusion is an open source implementation of the Latent Diffusion architecture, trained to denoise random Gaussian noise, in a lower dimensional latent space, to get a sample of interest.

Diffusion models are trained to predict a way to slightly denoise a sample in each step, and after a few iterations, a result is obtained. Diffusion models have already been applied to a variety of generation tasks, such as image, speech, 3D shape, and graph synthesis.



Diffusion models consist of two steps:
Forward Diffusion — Maps data to noise by gradually perturbing the input data. This is formally achieved by a simple stochastic process that starts from a data sample and iteratively generates noisier samples using a simple Gaussian diffusion kernel. This process is used only during training and not on inference. Nonetheless, it can be done faster using the following closed-form formula to directly get the noisy image at a specific time step t:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \varepsilon$$

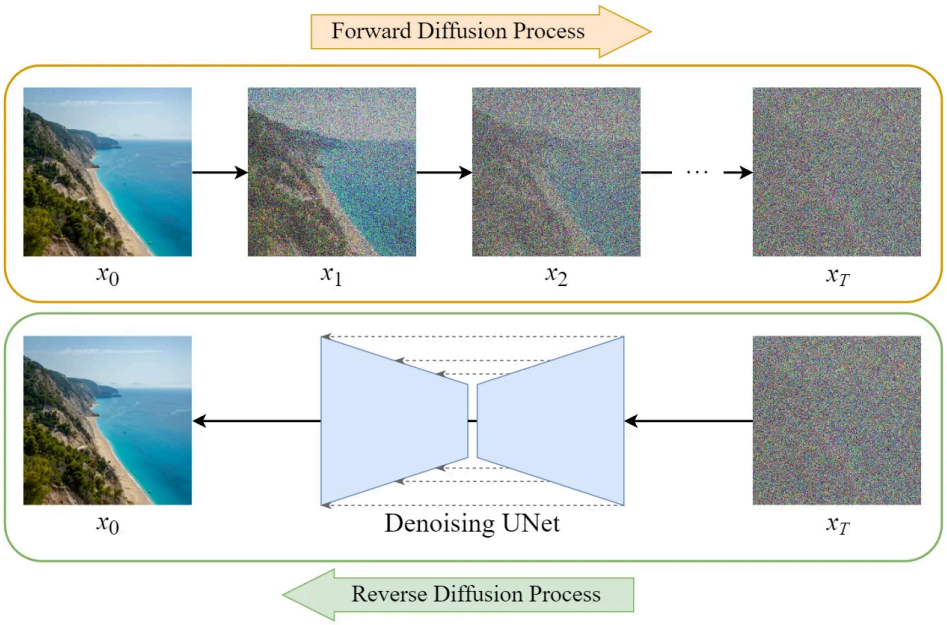
Parametrized Reverse — Undoes the forward diffusion and performs iterative denoising. This process represents data synthesis and is trained to generate data by converting random noise into realistic data. Since the reverse diffusion process is not directly computable, we train a neural network εθ to approximate it.

The training objective (loss function) is as follows:

$$x_t = \sqrt{\bar{a}_t}\, x_0 + \sqrt{1 - \bar{a}_t}\, \varepsilon$$

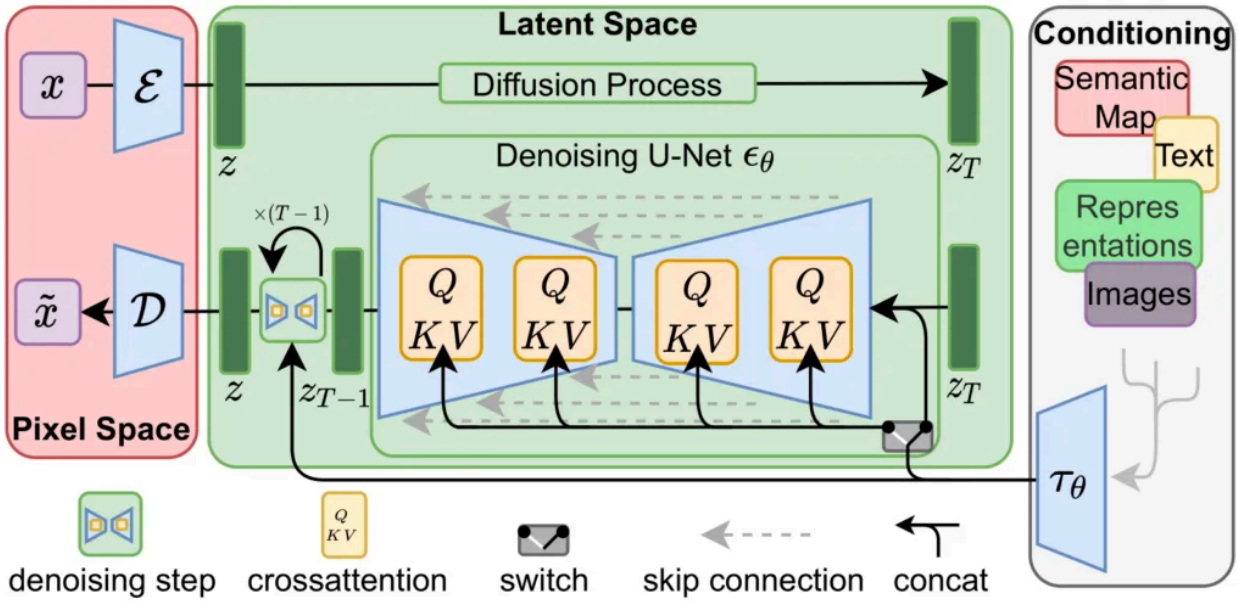
$$L_{\text{simple}} = \mathbb{E}_{t,x_0,\boldsymbol{\varepsilon}}\left[\|\boldsymbol{\varepsilon} - \boldsymbol{\varepsilon}_\theta(x_t, t)\|^2\right]$$

Training objective for the diffusion model:



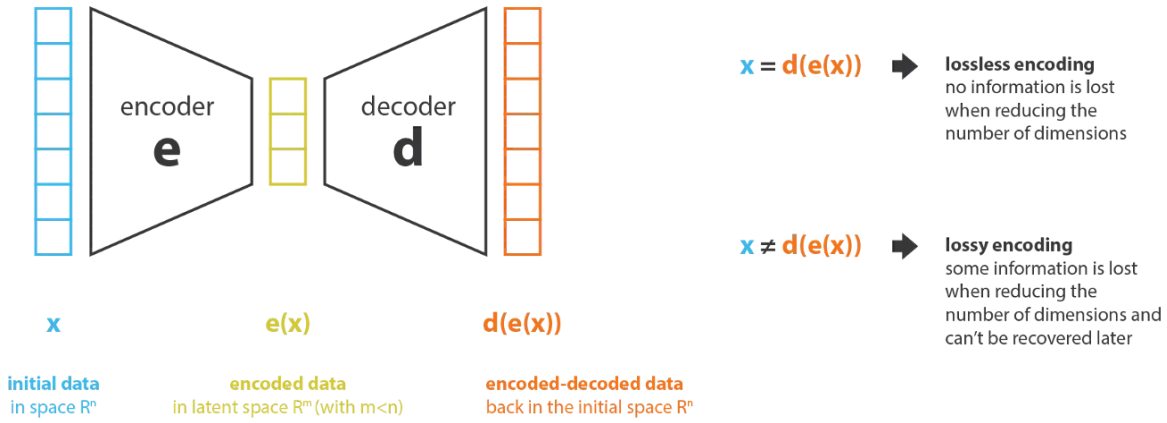
The forward and reverse processes require sequential repetition of thousands of steps, injecting and reducing noise, which makes the whole process slow and heavy on computational resources.

To enable training on limited resources while retaining its quality and flexibility, the creators of Stable Diffusion adopted the method suggested in the paper. Instead of using the actual pixel space, they applied the diffusion process over a lower dimensional latent space.



Stable Diffusion Architecture:
The Stable Diffusion architecture has three main components, two for reducing the sample to a lower dimensional latent space and then denoising random gaussian noise, and one for text processing.

1) **The Autoencoder:** The input of the model is a random noise of the size of the desired output. It will first reduce the sample to a lower dimensional latent space. For that, the authors used the VAE Architecture, which consists of two parts — encoder and decoder. The encoder is used during training to convert the sample into a lower latent representation and passes it as input to the next block. On inference, the denoised, generated samples undergo reverse diffusion and are transformed back to their original dimensional latent space.



2) **U-Net:** The U-Net block, composed of ResNet, receives the noisy sample in a lower latency space, compresses it, and then decodes it back with less noise. The estimated noise residual from the U-Net output is used to construct the expected denoised sample representation.

3) **Text Encoder:** The text encoder is responsible for the text processing, transforming the prompt into an embedding space. Similar to Google's Imagen, Stable Diffusion uses a frozen CLIP ViT-L/14 Text Encoder.

Training:

Training Data The model developers used the following dataset for training the model:

- LAION-5B and subsets thereof (see next section)

Training Procedure Stable Diffusion is a latent diffusion model which combines an autoencoder with a diffusion model that is trained in the latent space of the autoencoder. During training,

- Images are encoded through an encoder, which turns images into latent representations. The autoencoder uses a relative downsampling factor of 8 and maps images of shape H x W x 3 to latents of shape H/f x W/f x 4
- Text prompts are encoded through a ViT-L/14 text-encoder.
- The non-pooled output of the text encoder is fed into the UNet backbone of the latent diffusion model via cross-attention.
- The loss is a reconstruction objective between the noise that was added to the latent and the prediction made by the UNet.

Applications of Diffusion Models:

- Image Generation: Create high-quality, photorealistic images from scratch.
- Image Editing: Enhance existing images by adding, removing, or modifying specific elements.
- Video Generation: Generate realistic videos or manipulate existing videos by adding or removing objects or scenes.
- Text-to-Image Synthesis: Create images based on textual descriptions.

Code:

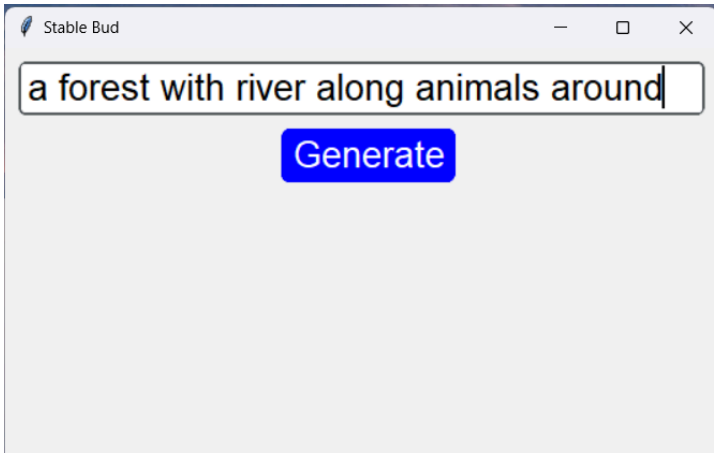
```
import tkinter as tk
import customtkinter as ctk

from PIL import ImageTk
from authtoken1 import auth_token

import torch
from torch import autocast
```

```
from diffusers import StableDiffusionPipeline

auth_token='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'
app = tk.Tk()
app.geometry("532x632")
app.title("Stable Bud")
ctk.set_appearance_mode("dark")

prompt = ctk.CTkEntry(height=40, width=512, text_font=("Arial", 20), text_color="black",
fg_color="white")
prompt.place(x=10, y=10)

lmain = ctk.CTkLabel(height=512, width=512)
lmain.place(x=10, y=110)

modelid = "CompVis/stable-diffusion-v1-4"
device = "cuda"
pipe = StableDiffusionPipeline.from_pretrained(modelid, revision="fp16",
torch_dtype=torch.float16, use_auth_token=auth_token)
pipe.to(device)

def generate():
    with autocast(device):
        image = pipe(prompt.get(), guidance_scale=8.5)["sample"][0]

    image.save('generatedimage.png')
    img = ImageTk.PhotoImage(image)
    lmain.configure(image=img)

trigger = ctk.CTkButton(height=40, width=120, text_font=("Arial", 20), text_color="white",
fg_color="blue", command=generate)
trigger.configure(text="Generate")
trigger.place(x=206, y=60)

app.mainloop()
```

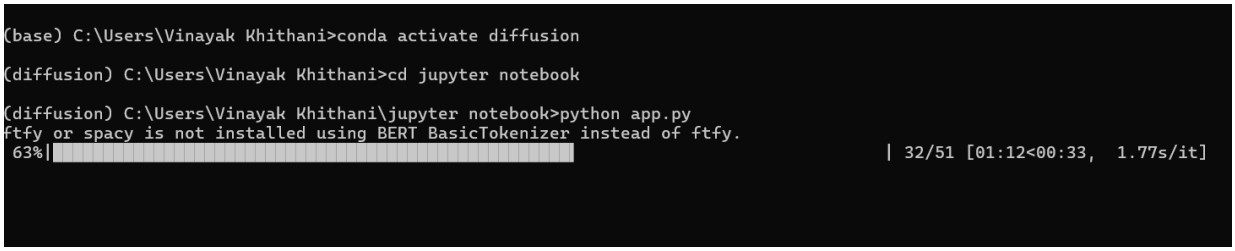
The above code is a Python application using Tkinter for GUI. It allows users to input text prompts, which are then processed through a Stable Diffusion model (likely for image generation) using PyTorch. The generated image is displayed on the GUI. The user triggers the generation process by clicking the "Generate" button.

Model Execution:

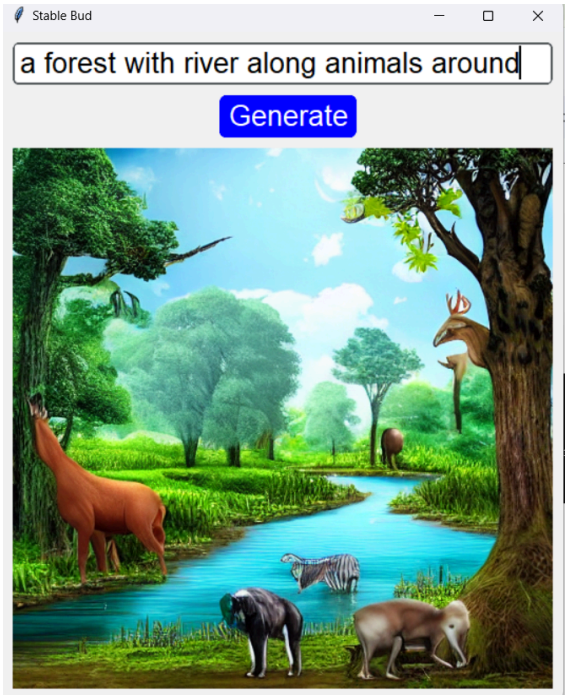
1. Tkinter GUI

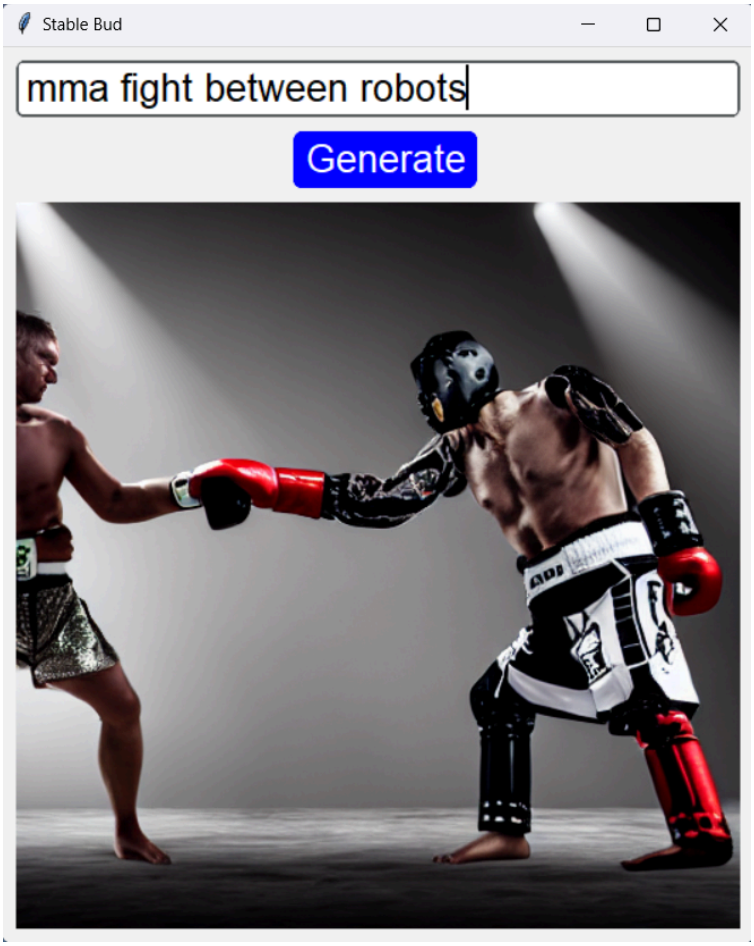


2. Epochs running



3. Output:





Limitations and Bias

Limitations:

- The model does not achieve perfect photorealism.
- The model cannot render legible text.
- The model does not perform well on more difficult tasks which involve compositionality, such as rendering an image corresponding to “A red cube on top of a blue sphere”.
- Faces and people in general may not be generated properly.
- The model was trained mainly with English captions and will not work as well in other languages.
- The autoencoding part of the model is lossy
- The model was trained on a large-scale dataset LAION-5B which contains adult material and is not fit for product use without additional safety mechanisms and considerations.
- No additional measures were used to deduplicate the dataset. As a result, we observe some degree of memorization for images that are duplicated in the training data. The training data can be searched at https://rom1504.github.io/clip-retrieval/ to possibly assist in the detection of memorized images.

Bias:
While the capabilities of image generation models are impressive, they can also reinforce or exacerbate social biases. Stable Diffusion v1 was primarily trained on subsets of LAION-2B(en), which consists of images that are limited to English descriptions. Texts and images from communities and cultures that use other languages are likely to be insufficiently accounted for. This affects the overall output of the model, as white and western cultures are often set as the default. Further, the ability of the model to generate content with non-English prompts is significantly worse than with English-language prompts. Stable Diffusion v1 mirrors and exacerbates biases to such a degree that viewer discretion must be advised irrespective of the input or its intent.

References:
https://github.com/CompVis/stable-diffusion/tree/main
https://github.com/nicknochnack/StableDiffusionApp/tree/main
https://towardsdatascience.com/stable-diffusion-best-open-source-version-of-dall-e-2-ebcdf1cb64bc