MUFFAKHAM JAH

COLLEGE OF ENGINEERING & TECHNOLOGY

(THE SULTAN UL ULOOM EDUCATION SOCIETY)

Affiliated to Osmania University & Recognized by AICTE

Banjara Hills, Hyderabad 500 034

DEPARTMENT OF COMPUTER SCIENCE & ARTIFICIAL INTELLIGENCE

EXPOGENIX ‘23

TEXT2IMAGE

TEAM

Mohammed Junaid Adil (1604-20-747-014)

Mohammed Ahmed Hussain (1604-20-747-019)

Mohammed Sahil Arman (1604-20-747-053)

21-01-2023

Abstract

Text-to-image generation has traditionally focused on finding better modeling assumptions for training on a fixed dataset. These assumptions might involve complex architectures, auxiliary losses, or side information such as object part labels or segmentation masks supplied during training. By decomposing the image formation process into a sequential application of denoising autoencoders, diffusion models (DMs) achieve state-of-the-art synthesis results on image data and beyond. These models typically operate directly in pixel space, optimization of powerful DMs often consumes hundreds of GPU days and inference is expensive due to sequential evaluations. To enable DM training on limited computational resources while retaining their quality and flexibility, we apply them in the latent space of powerful pretrained autoencoders. Training diffusion models on such a representation allows for the first time to reach a near-optimal point between complexity reduction and detail preservation. Latent diffusion models (LDMs) achieve new state-of-the-art scores for image inpainting and class-conditional image synthesis and highly competitive performance on various tasks, including text-to-image synthesis, while significantly reducing computational requirements compared to pixel-based DMs.

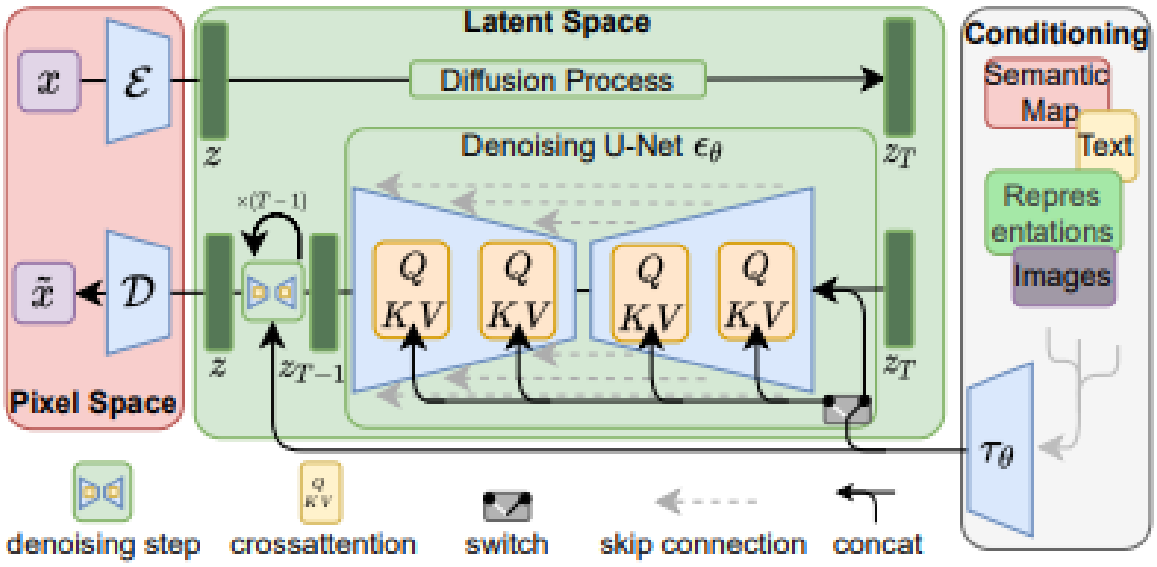
Introduction

A text-to-image model is a machine learning model which takes as input a natural language description and produces an image matching that description. Such models began to be developed in

the mid-2010s, as a result of advances in deep neural networks. In 2022, the output of state-of-the-art text-to-image models, such as OpenAI's DALL-E 2. The most effective models have generally been trained on massive amounts of image and text data scraped from the web. Stable Diffusion is a latent text-to-image diffusion model capable of generating photo-realistic images given any text input. It is a Latent Diffusion Model that uses a fixed, pretrained text encoder (CLIP ViT-L/14).

Methodology

The language model creates an embedding of the text prompt. It's fed into the diffusion model together with some random noise. The diffusion model denoises it towards the embedding. This is repeated several times. Then in the end the decoder scales the image up to a larger size.



The image generator goes through two stages:

1- **Image information creator**

This component is the secret sauce of Stable Diffusion. It's where a lot of the performance gain over previous models is achieved.

This component runs for multiple steps to generate image information. This is the *steps* parameter in Stable Diffusion interfaces and libraries which often defaults to 50 or 100.

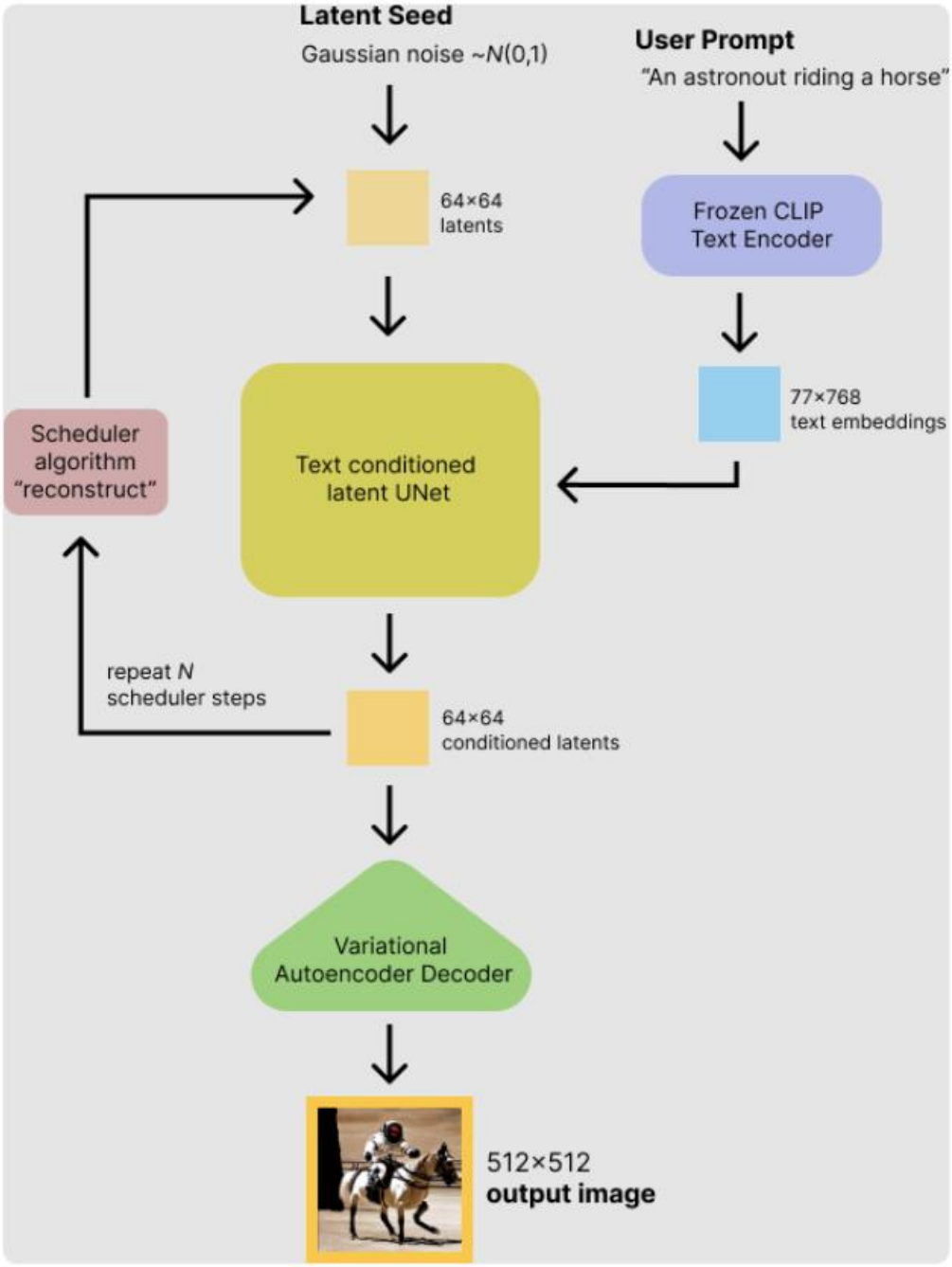
The image information creator works completely in the *image information space* (or *latent* space). We'll talk more about what that means later in the post. This property makes it faster than previous diffusion models that worked in pixel space. In technical terms, this component is made up of a UNet neural network and a scheduling algorithm.

The word "diffusion" describes what happens in this component. It is the step-by-step processing of information that leads to a high-quality image being generated in the end (by the next component, the image decoder).

2- **Image Decoder**

The image decoder paints a picture from the information it got from the information creator. It runs only once at the end of the process to produce the final pixel image.

With this we come to see the three main components (each with its own neural network) that make up Stable Diffusion:



1. The autoencoder (VAE)

The VAE model has two parts, an encoder and a decoder. The encoder is used to convert the image into a low dimensional latent representation, which will serve as the input to the *U-Net* model.

The decoder, conversely, transforms the latent representation back into an image.

During latent diffusion *training*, the encoder is used to get the latent representations (*latents*) of the images for the forward diffusion process, which applies more and more noise at each step. During *inference*, the denoised latents generated by the reverse diffusion process are converted back into images using the VAE decoder. As we will see during inference, we **only need the VAE decoder**.

2. The U-Net

The U-Net has an encoder part and a decoder part both comprised of ResNet blocks. The encoder compresses an image representation into a lower resolution image representation and the decoder decodes the lower resolution image representation back to the original higher resolution image representation that is supposedly less noisy. More specifically, the U-Net output predicts the noise residual which can be used to compute the predicted denoised image representation.

To prevent the U-Net from losing important information while downsampling, short-cut connections are usually added between the downsampling ResNets of the encoder to the upsampling ResNets of the decoder. Additionally, the stable diffusion U-Net is able to condition its output on text-embeddings via cross-attention layers. The cross-attention layers are added to both the encoder and decoder part of the U-Net usually between ResNet blocks.

3. The Text-encoder

The text-encoder is responsible for transforming the input prompt, *e.g.* "An astronaut riding a horse" into an embedding space that can be understood by the U-Net. It is usually a simple *transformer-based* encoder that maps a sequence of input tokens to a sequence of latent text-embeddings.

Inspired by Imagen, Stable Diffusion does **not** train the text-encoder during training and simply uses an CLIP's already trained text encoder, CLIPTextModel.

Implementation

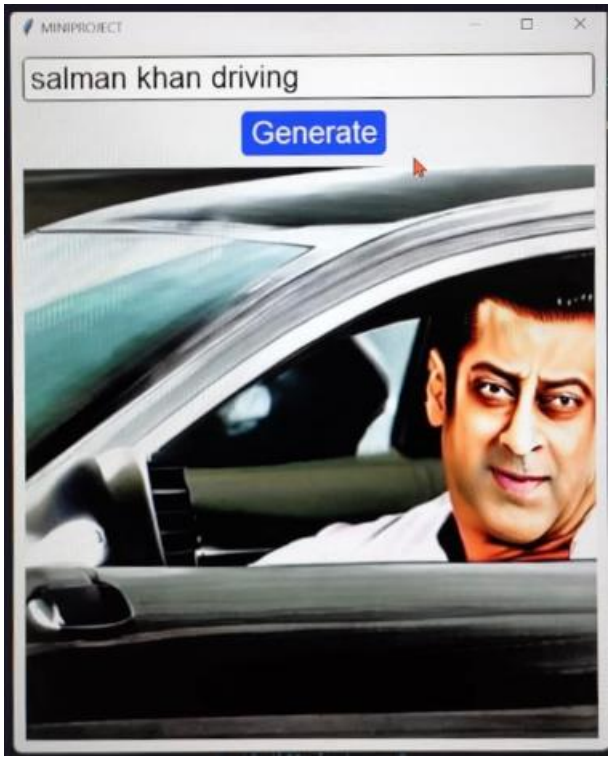
Software requirements:

- Python 3.9
- Python libraries: PyTorch, Tkinter, CustomTkinter, BERT Tokenizer, Diffusers.
- VS Code

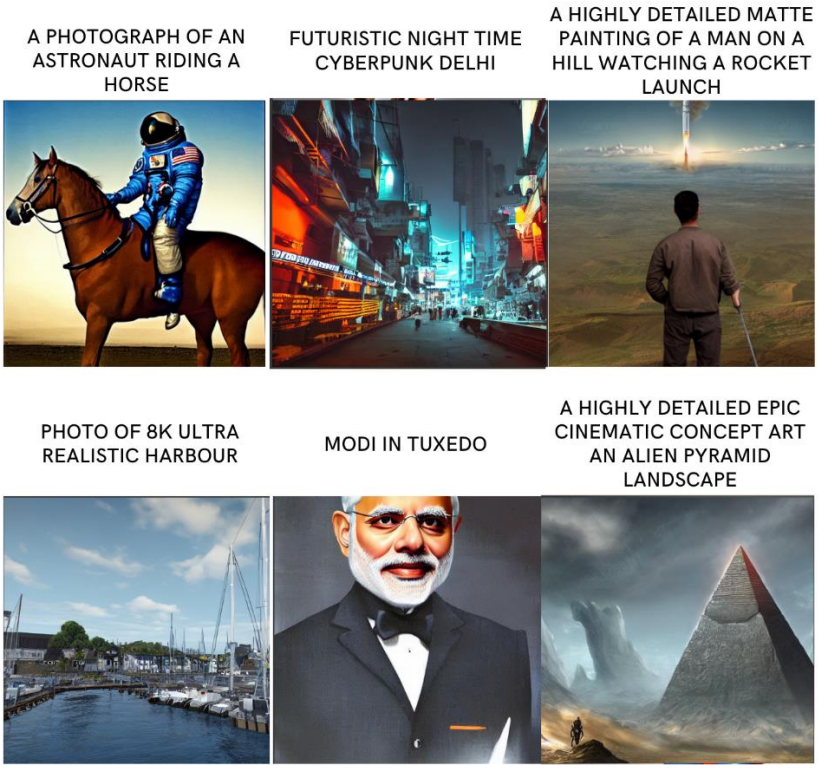
Hardware requirements:

- A modern Intel or AMD CPU.
- 8 GB RAM
- A GPU with the video memory of 6 GB of above.

User Interface



Results



Conclusion

Latent diffusion models, a simple and efficient way to significantly improve both the training and sampling efficiency of denoising diffusion models without degrading their quality. Based on this and a cross-attention conditioning mechanism, this experiments could demonstrate favorable results compared to state-of-the-art methods across a wide range of conditional image synthesis tasks without task-specific architectures.

Reference

High-Resolution Image Synthesis with Latent Diffusion Models - Rombach, R., Blattmann, A., Lorenz, D., Esser, P. and Ommer, B., 2022. High-resolution image synthesis with latent diffusion models. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition

Zero-Shot Text-to-Image Generation - Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, Ilya Sutskever

Learning Transferable Visual Models From Natural Language Supervision - Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, Ilya Sutskever

Hierarchical Text-Conditional Image Generation with CLIP Latents - Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, Mark Chen